UNIVERSITÉ PARIS-SUD XI

ÉCOLE DOCTORALE D'INFORMATIQUE

THÈSE DE DOCTORAT

Soutenue le 19 novembre 2012 par

Rémi Bardenet

Towards adaptive learning and inference

Applications to hyperparameter tuning and astroparticle physics

préparée au Laboratoire de l'Accélérateur Linéaire et au Laboratoire de Recherche en Informatique de l'Université Paris-Sud XI, dans les équipes AppStat, Auger et Tao

sous la direction de Balázs Kégl

Jury

Président	Arnaud Doucet	University of Oxford
Rapporteurs	Éric Moulines	Télécom ParisTech
	Christian Robert	Université Paris-Dauphine, IUF & CREST
Directeur	Balázs Kégl	Université Paris-Sud & CNRS
Examinateurs	Francis Bach	INRIA & ENS Paris
	Fabien Cavalier	Université Paris-Sud
Invité	Radek Stompor	Université Paris-Diderot & CNRS

Last modified on November 21, 2012

Acknowledgments
Remerciements

I would like to thank my advisor, Balázs Kégl, who is not only an inspired and inspiring scientist, but also a smart manager of people.

I owe a great deal to regular collaborators Gersende Fort and Olivier Cappé from Telecom ParisTech, who taught me much more than technical skills.

My thanks go also to the Auger LAL team, especially to Karim louedec and Pham Ngoc Diep who helped me integrate the world of particles. More generally, thanks to Auger members all around the world, among which I made good friends while eating tender *bifes de chorizo*.

Thanks also to all TAO members for fruitful discussions, especially Niko Hansen, Tamás Eltetö, Cécile Germain-Renaud, Michèle Sebag, and Yann Ollivier. This work has also benefited from exchanges with Julien Bect and Alireza Roodaki at Supélec.

I have a special and friendly thought for all the past and present members of the AppStat team at LAL, Robi, Djalel, Matyi, FD, and Diego, keep up the good work in this warm atmosphere, now powered with tasty, gradually cheaper coffee.

Thanks to Yoshua Bengio for welcoming me at UdeM in Montréal, to James Bergstra for the good work together and to everybody at LISA for the good times there.

Thanks also to everyone that has been involved in the teaching part of my thesis, at Université Paris-Sud and Polytech' Paris-Sud, among others Michel Menou, Jérôme Azé, Claude Barras, Cédric Bentz, and Aurélien Max, for sharing their experience.

Finally, let me thank the staff at LAL, the efficient *service missions* and *secrétariat*, and especially Françoise Maréchal at the library. Thanks also to Stéphanie Druetta at the computer science doctoral school. These people have contributed to the great working environment I enjoyed during my thesis.

Thanks also to my reviewers and jury members for kindly accepting to participate to the final step of this thesis.

Enfin, un énorme merci à Amélie, qui est toujours à mes côtés, derrière moi ou sur mon dos, selon les besoins, à mes parents, dont la présence rassurante me donne des ailes, et à tous mes amis qui sont heureusement trop nombreux pour être énumérés ici.

Summary in French
Résumé en français

Nous proposons ici une introduction en français aux problèmes traités dans cette thèse ainsi qu'un résumé des contributions qui y sont présentées.

En apprentissage artificiel, ou ML pour *machine learning*, le processus d'apprentissage d'un algorithme correspond généralement à deux boucles imbriquées. La boucle externe itère sur les valeurs des *hyperparamètres*, alors que la boucle interne minimise une mesure d'erreur empirique. Pour fixer les idées, considérons un problème de classification auquel on applique une machine à vecteurs support (SVM, voir [27, 46]) avec un noyau gaussien isotrope à un paramètre, appelé longueur caractéristique. Les deux hyperparamètres de l'algorithme sont alors cette longueur caractéristique et le coefficient multiplicatif de la norme du classifieur dans l'objectif de la SVM. Traditionnellement, on fixe ensuite une grille bidimensionnelle couvrant une partie de l'espace des hyperparamètres, et pour chaque point de cette grille, on exécute la procédure de maximisation de la marge de la SVM – la boucle interne –, en retournant une erreur de validation. Finalement, on choisit dans la grille la valeur des hyperparamètres qui a retourné la plus petite erreur de validation. Il est également commun que l'utilisateur intervienne pendant la boucle externe et affine la grille dans des régions « prometteuses » de l'espace des hyperparamètres.

Si on prend l'exemple de la classification, un algorithme d'apprentissage est une fonction qui associe un classifieur à un jeu de données. Cette fonction inclut un budget spécifiant d'une part combien de cycles CPU sont à consacrer à l'ajustement des hyperparamètres – la boucle externe – et combien de cycles sont à consacrer à l'évaluation de chaque choix des hyperparamètres – la boucle interne. Des résultats récents comme [103] et [43] suggèrent qu'avec le matériel d'aujourd'hui, en particulier les structures de calcul massivement parallèles, une allocation optimale de ce budget devrait favoriser l'exploration des hyperparamètres plus nettement qu'à l'accoutumée.

Plusieurs arguments plaident en faveur du développement de méthodes d'ajustement des hyperparamètres plus efficaces et, surtout, plus automatiques que la recherche exhaustive à base de grilles et d'interventions de l'utilisateur. Tout d'abord, la boucle interne d'un algorithme d'apprentissage correspond souvent à un problème d'optimisation en grande dimension, dont la résolution requiert l'exécution d'algorithmes gloutons en évaluations de la fonction objectif, comme les variantes de la descente de gradient. Appliquée aux bases de données énormes qui sont monnaie courante aujourd'hui, une seule itération de la boucle externe est dès lors une question d'heures, voire de jours. Par ailleurs, des algorithmes constituant l'état de l'art en classification comme les réseaux de neurones profonds [20] possèdent des dizaines d'hyperparamètres, ce qui rend la recherche exhaustive infaisable. De plus, de récentes études [103, 41, 43] montrent que le difficile problème de l'ajustement des hyperparamètres dans les réseaux profonds est un obstacle majeur au progrès scientifique. Ces études ont dépassé l'état de l'art en classification d'images grâce à un meilleur ajustement des hyperparamètres de modèles simples, plutôt que par des modélisations innovantes ou des stratégies d'apprentissage radicalement nouvelles. Enfin, la disponibilité de solutions logicielles pour l'ajustement automatique des hyperparamètres en apprentissage permettrait d'une part une utilisation simplifiée pour les non-experts, et donc de décupler l'impact des découvertes en ce domaine où les applications sont nombreuses ; d'autre part, l'utilisation généralisée d'une même méthode automatique rendrait également plus juste la comparaison de résultats expérimentaux proposés dans des études différentes.

Considérons maintenant un autre domaine où des questions similaires se posent. En statistiques computationnelles, les méthodes dites MCMC, pour *Markov chain Monte Carlo*, constituent une approche générique à l'échantillonnage de distributions de probabilité complexes. Les méthodes MCMC sont aujourd'hui *de facto* l'outil standard de nombreuses applications en inférence bayésienne. L'algorithme MCMC le plus répandu est l'algorithme de Metropolis-Hastings (MH, voir [110]). MH construit une chaîne de Markov qui approxime des tirages indépendants identiquement distribués selon sa cible. En d'autres termes, MH est une boucle qui, à chaque itération, propose un mouvement aléatoire local selon une *distribution de proposition*, et accepte ou rejette le point proposé selon une règle sophistiquée. Le choix de la distribution de proposition est crucial pour obtenir des résultats satisfaisants [110]. Lorsqu'on applique MH à des distributions cibles de petite dimension, il est courant d'ajuster la distribution de proposition lors d'une étude préliminaire, en essayant d'optimiser des critères variés. Cependant, lorsque qu'on a affaire à des algorithmes MCMC complexes, parfois imbriqués les uns dans les autres, et à des cibles de haute dimension, comme les distributions a posteriori dans des modèles génératifs à grande échelle, les interventions manuelles se compteraient en millions, et il n'est pas même évident de savoir quel critère utiliser pour ajuster la distribution de proposition. De tels modèles génératifs à grande échelle sont en particulier courants en physique des particules : l'expérience Pierre Auger, par exemple, est une expérience actuelle dévolue à l'étude des rayons cosmiques, synonyme de près de 1600 détecteurs et de plusieurs centaines de milliers de signaux.

Le but de cette thèse est de rendre l'application des algorithmes d'apprentissage et d'inférence plus *autonomes* en leur faisant *apprendre* la structure du problème qu'ils sont appelés à résoudre. Dans le contexte de l'ajustement des hyperparamètres en apprentissage, l'optimisation séquentielle à base de modèles (SMBO, pour *sequential model-based optimization*, voir [82]), aussi appelée optimisation bayésienne, a récemment suscité un vif intérêt [21, 126, 120]. Contrairement aux méthodes de la famille des descentes de gradient, SMBO est un paradigme particulièrement adapté aux situations où l'évaluation d'un vecteur d'hyperparamètres est coûteuse. A chaque itération, SMBO profite de l'information disponible à partir des précédentes évaluations pour construire un modèle de la fonction objectif, ici une erreur de validation, et utilise ce modèle pour choisir le prochain point à évaluer. Cette dernière tâche est effectuée en optimisant un critère auxiliaire qui mesure l'intérêt de l'évaluation de chaque point, connaissant le modèle courant de la fonction objectif. A moyen terme, nous imaginons que l'exécution d'algorithmes d'apprentissage se fera de façon complètement automatique, ne requérant aucun ajustement manuel de la part de l'utilisateur. Nous pensons que SMBO est un environnement idéal pour développer cette automatisation, et nous présentons dans la première partie de cette thèse les contributions que nous avons apportées pour rendre cette vision accessible.

Tout d'abord, le chapitre 2 présente SMBO, ainsi que notre contribution à l'étape d'optimisation du critère auxiliaire. Ces critères, le plus connu étant certainement l'amélioration en espérance (EI, pour *expected improvement*, voir [82]), sont habituellement optimisés par une recherche exhaustive dans l'espace des hyperparamètres, à l'aide de grilles, d'échantillons aléatoires ou pseudoaléatoires. La figure 2.2 illustre le critère EI – en rouge – lorsque le modèle de la fonction objectif est un processus gaussien [108] – en bleu. Nous avons proposé dans [14] un algorithme d'optimisation évolutionnaire dédié à l'optimisation de tels critères qui utilise ce qu'on connaît de leur forme : en travaillant l'expression d'EI, on réalise qu'EI est nul aux points où la fonction objectif a déjà été évaluée, et est multimodal. Nous montrons au chapitre 2 que cette approche améliore sensiblement l'optimisation d'EI en pratique. Ensuite, pour montrer que SMBO est de taille à s'attaquer aux défis actuels du ML, nous présentons au chapitre 4 des applications sur mesure de SMBO à l'ajustement des plus de 30 hyperparamètres de réseaux de neurones profonds (DBNs, pour *deep belief networks*), que nous avons publiées dans [21]. Nous démontrons premièrement que, de façon surprenante, la méthode consistant simplement à tirer un échantillon aléatoire de vecteurs d'hyperparamètres qu'on évalue ensuite donne de meilleurs résultats sur cette tâche que l'ajustement manuel. Ensuite, nous démontrons que SMBO, à son tour, est plus efficace que ces deux approches. Motivés par ces résultats positifs, nous avons réalisé qu'un des avantages cruciaux que les humains avaient encore sur les algorithmes d'ajustement automatique des hyperparamètres était leur mémoire des expériences passées avec des jeux de données similaires. Un utilisateur qui a déjà trouvé dans le passé de bons hyperparamètres à son algorithme pour des données présentant des caractères communs avec les données qui l'intéressent actuellement (comme la taille de l'échantillon, le nombre d'attributs, ou des propriétés statistiques décrivant la forme de l'échantillon) aura certainement pour intuition d'évaluer ces mêmes anciens hyperparamètres sur ses nouvelles données, ou encore de considérer certaines zones de l'espace des hyperparamètres comme prometteuses *a priori*. Dans le chapitre 5, nous détaillons la construction, non encore publiée à la date de soutenance de cette thèse, d'un algorithme nommé SCoT (pour *surrogate collaborative tuning*), qui implémente SMBO sur l'espace produit des hyperparamètres et des jeux de données, permettant ainsi de traduire cette intuition humaine. SCoT peut, par exemple, *(i)* tirer parti de l'ajustement simultané du même algorithme sur des jeux de données différents, et surtout *(ii)* utiliser l'information gagnée dans le passé par l'ajustement du même algorithme sur d'autres jeux de données. La principale difficulté méthodologique apparaît lorsqu'on souhaite comparer les résultat de l'algorithme en question sur des jeux de données différents : les traditionnelles erreurs de validation ne peuvent être utilisées directement, puisque deux jeux de données différents peuvent conduire à des erreurs à des échelles radicalement différentes. La cible de l'algorithme SMBO est donc ici plus difficile à définir que dans nos applications précédentes. SCoT résout ce problème en faisant appel à chaque itération à un algorithme de *ranking* qui construit un modèle d'une fonction latente d'amplitude constante à travers les différents jeux de données, et qui, pour chaque jeu de données, est une fonction croissante de l'erreur de validation sur ce jeu. Nous présentons une application de SCoT à AdaBoost, un algorithme de classification populaire. SCoT est également intéressant d'un point de vue méthodologique, puisque c'est un algorithme SMBO générique et nouveau, en ce qu'il adapte sa fonction objectif en ligne.

Revenons maintenant aux algorithmes MCMC. L'ajustement de la distribution de proposition de l'algorithme de Metropolis-Hastings (MH) peut être comparé au choix des hyperparamètres d'un algorithme d'apprentissage. Depuis l'article fondateur [69], les méthodes MCMC *adaptatives* qui apprennent leur distribution de proposition en même temps qu'elles échantillonnent, et donc amassent de l'information sur leur cible, sont devenues l'objet d'un domaine de recherche dédié. L'algorithme Metropolis adaptatif (AM, voir [69], par exemple, estime la matrice de covariance de sa distribution cible pendant l'exécution d'un MH avec distribution de proposition gaussienne multivariée. La motivation principale de cette approche vient de résultats théoriques [111, 112] qui suggèrent que la corrélation de la chaîne produite par l'algorithme est minimisée lorsque la covariance de la proposition correspond à celle de la cible, et ce pour une large classe de distributions unimodales aux marginales indépendantes. AM calque donc la covariance de sa distribution de proposition sur l'estimation qu'elle fait, à chaque itération et en utilisant un schéma d'approximation stochastique, de la covariance de sa cible. AM est un exemple d'algorithme MCMC avec ajustement automatique de ses hyperparamètres, qui fonctionne en utilisant l'information amassée sur sa cible au fur et à mesure de son exécution. Cette remarque permet de rapprocher AM des méthodes SMBO d'ajustement des hyperparamètres d'algorithmes d'apprentissage dont nous avons parlé plus haut. La partie II de cette thèse est consacrée aux algorithmes MCMC adaptatifs et à leur application.

Notre intérêt pour les algorithmes MCMC adaptatifs a été initialement motivé par notre implication dans l'analyse des données de l'expérience Pierre Auger, que nous abrégerons ici en *Auger*. Portant le nom d'un physicien français du début du vingtième siècle, Auger est un observatoire géant de rayons cosmiques qui est présenté au chapitre 6. Les rayons cosmiques sont des particules chargées qui voyagent dans l'univers à des énergies particulièrement élevées. Lorsqu'une de ces particules rencontre les molécules de gaz de notre atmosphère, elle génère une cascade de particules, parmi lesquelles des *muons*, qui arrosent la surface de la Terre sur plusieurs kilomètres carrés. Auger possède, entre autres, un réseau de détecteurs au sol couvrant plus de 3 000 km^2 de pampa argentine et collectant des données sur ces larges cascades. L'objectif de l'analyse des données est d'estimer, à partir des données d'une cascade, certains paramètres de la première particule qui a frappé l'atmosphère. Déterminer la composition chimique des rayons cosmiques, par exemple, aurait d'importantes conséquences en astrophysique, comme détaillé dans le chapitre 6. Nous avons donc commencé la dérivation d'un modèle génératif complet d'Auger qui permette de pratiquer l'inférence bayésienne des paramètres de chaque cascade. Dans le chapitre 7, nous décrivons la première partie de ce modèle, que nous présentâmes en premier dans [17], et qui correspond au signal généré par les muons dans un unique détecteur. Dans la seconde moitié du chapitre 7, nous exposons une procédure dite « bayésienne empirique » qui requiert uniquement ce premier modèle pour donner des résultats sur les données d'Auger.

Les algorithmes adaptatifs comme AM sont particulièrement efficaces pour l'inférence dans des modèles complexes et de grande dimension comme celui d'Auger. Cependant, le modèle que nous décrivons au chapitre 7 est un modèle additif formé de la somme de plusieurs composantes, une par muon, et qui est invariant aux permutations de ces composantes. Cette invariance rend l'inférence bayésienne sujette à des difficultés d'étiquetage des composantes, connues en anglais sous le nom de *label switching* : la loi a posteriori des paramètres de toutes les composantes possède un nombre exponentiel de modes redondants, un par permutation des composantes, ce qui rend les marginales issues d'un algorithme MCMC difficiles à interpréter. De plus, l'utilisation de propositions MCMC adaptatives est rendue impossible par la non-identifiabilité des composantes du modèle. Le chapitre 8 présente une revue des méthodes existantes pour résoudre le *label switching*. Aucune de ces méthodes n'est à la fois *(i)* théoriquement justifiée, *(ii)* indépendante de l'expertise de l'utilisateur et *(iii)* compatible avec AM. Ce vide méthodologique nous a inspiré AMOR [14], un algorithme de Metropolis adaptatif avec rééitiquetage en ligne qui possède ces trois propriétés. Nous présentons AMOR au chapitre 9. Nous illustrons ensuite AMOR sur un exemple qui met en lumière ses différences par rapport aux approches existantes qui lui sont apparentées. Nous appliquons également AMOR à des tâches d'inférence bayésienne comme l'estimation des paramètres de modèles de mélange, un exemple courant possédant la même invariance par symétrie qui a motivée AMOR, ou comme l'inférence des paramètres individuels de chaque muon dans notre modèle d'Auger. Le premier exemple est tiré d'un article que nous avons soumis récemment [13], les deux derniers sont apparus respectivement dans [14] et [16].

Le chapitre 10 est dédié à l'étude théorique de la convergence d'AMOR, que nous avons également soumise dans [13]. L'un des défis techniques est qu'AMOR est adaptatif non seulement en ses propositions, mais aussi en sa cible. En effet, nous montrons qu'AMOR apprend une restriction optimale, en un sens que nous précisons, de la cible originelle à un de ses modes redondants. Nos résultats de convergence constituent la première analyse théorique d'un algorithme de rééitiquetage en ligne à notre connaissance. Le cadre présenté possède les avantages de *(i)* généraliser certaines approches préexistantes, *(ii)* fournir une caractérisation explicite de la loi a posteriori obtenue après réétiquetage, *(iii)* paver la voie vers une future étude du comportement asymptotique des algorithmes de réétiquetage, et enfin *(iv)* il met en lumière un lien clair et élégant entre le réétiquetage et la quantification vectorielle, qui était jusque là seulement subodoré.

Finalement, dans le chapitre 11, nous revenons sur des liens et questions esquissés dans le texte principal. Tout d'abord, nous présageons une unification prochaine des algorithme de bandits, de SMBO et d'autres approches similaires décrites au chapitre 3. Cette unification est déjà en marche, au travers d'articles comme [122] où le critère auxiliaire de SMBO est inspiré par l'algorithme UCB [11], ou encore les algorithmes de bandits bayésiens de [83]. Les cadres de SMBO et des bandits ont beaucoup à s'apporter mutuellement, le premier pouvant s'inspirer des garanties théoriques fournies par le second, et ce dernier pouvant tirer parti des avancées méthodologiques et expérimentales de la communauté SMBO. Nous discutons ensuite la possibilité d'un système d'ajustement des hyperparamètres qui soit commun à tous les utilisateurs du même algorithme et qui implémenterait SCoT avec un horizon temporel infini. Imaginons en effet que tous les utilisateurs du monde de perceptrons à une couche cachée se mettent d'accord sur les hyperparamètres à utiliser et sur une façon d'encoder les propriétés d'un jeu de données en un vecteur de réels. Chaque nouveau jeu de données serait chargé dans une base de données commune et l'utilisateur recevrait régulièrement un rapport sur l'ajustement de l'algorithme à son jeu de données. Pour qu'une telle utopie n'en soit plus une, il faudrait lever encore plusieurs difficultés méthodologiques. Premièrement, les utilisateurs ont en général une date limite à respecter, à laquelle ils doivent avoir obtenu les meilleurs résultats possibles. Des algorithmes d'ajustement des hyperparamètres prenant en compte cet horizon fini sont donc souhaitables. Deuxièmement, les processus gaussiens et leur complexité cubique en le nombre de points évalués ne sont pas un choix idéal de modèle si on envisage un grand nombre de jeux de données à traiter. D'un côté, des améliorations significatives ont déjà été apportées pour réduire ce coût, voir [108, Chapitre 8], et [38] pour une comparaison expérimentale récente de l'état de l'art. D'un autre côté, d'autres modèles que les processus gaussiens peuvent et ont déjà été introduits dans SMBO, comme les arbres de régression de [77], et nous pensons qu'il y a de la place pour d'autres méthodes de régression. Enfin nous ouvrons sur des réflexions concernant les liens entre AMOR et SCoT d'une part, et les algorithmes MCMC adaptatifs et les noyaux d'autre part.

Contents

Chapter 1
Introduction

In machine learning (ML), the training process of an algorithm generally consists of two nested loops: the outer loop iterates over *hyperparameter* values, while the inner loop minimizes an empirical error measure. For instance, assume we want to perform binary classification with a support vector machine (SVM; [27, 46]) and a kernel with a single parameter, such as the lengthscale of an isotropic, unit-amplitude squared exponential kernel. The hyperparameters are then this lengthscale and the coefficient of the norm penalty in the SVM objective function. Let us fix a finite two-dimensional grid on the hyperparameter space. For each point in the grid, we run the SVM margin maximization algorithm, and report a validation error. We finally pick the hyperparameter values that yield the smallest validation error among the grid points. It is also common practice for the user to intervene at some point in the outer loop and refine the grid in "promising" regions of the hyperparameter space. Overall, taking classification as an example, a learning algorithm is a *functional from data to classifier*, and includes a budgeting choice of how many CPU cycles are to be spent on hyperparameter exploration – the outer loop –, and how many CPU cycles are to be spent evaluating each hyperparameter choice – the inner loop. The results of [103] and [44] suggest that with current generation hardware such as large computer clusters and GPUs, the optimal allocation of CPU cycles includes more hyperparameter exploration than has been typical in the machine learning literature.

There are several arguments in favor of the development of more efficient and automatic approaches to hyperparameter tuning than manually assisted grid search. First, the inner loop of an ML algorithm often corresponds to a high-dimensional optimization problem that is solved using an algorithm that requires a large number of function evaluations, such as gradient descent variants. On today's large datasets, it is thus not unusual for one iteration of the outer loop to require hours or days. Second, state of the art classification algorithms such as deep belief networks (DBNs; [20]) have tens of hyperparameters, which makes grid search intractable. Furthermore, recent results such as [103], [42], and [44] demonstrate that the challenge of hyperparameter optimization in large multilayer models such as DBNs is a direct impediment to scientific progress. These works advanced state of the art performance on image classification problems by more concerted hyperparameter optimization in simple algorithms, rather than by innovative modelling or machine learning strategies. Third, automatic tuning software will make ML algorithms easier to use for non-expert users and allow experimental results from different studies to be compared more fairly.

We now consider another field, where similar questions arise. In computational statistics, Markov chain Monte Carlo (MCMC) is a generic approach for exploring complex probability distributions based on sampling. It has become the *de facto* standard tool in many applications of Bayesian inference. The most widely applicable MCMC algorithm is the Metropolis-Hastings algorithm (MH; [110]), which builds a Markov chain that approximates independent draws of the desired target distribution. MH is basically a loop which, at each iteration, makes a local random proposal according to a *proposal distribution*, and accepts or rejects the new point as its current state according to a sophisticated rule. Fine tuning of the proposal distribution is crucial to obtain good results [110]. When applying MH to sample from a small-dimensional target, it is customary to use rules of thumb or to set up the proposal distribution in a preliminary analysis, optimizing various criteria. However, when running imbricated MCMC algorithms on a complex high-dimensional target, such as the posterior distribution in a large-scale generative model, manual interventions would often have to be counted in millions, and it is not even obvious what the user should aim for. Such large-scale models are now common in particle physics: the Pierre Auger experiment, for instance, is a running experiment designed to study cosmic rays, and involves around 1600 detectors and several 100K signals.

The purpose of this thesis is to make the application of inference and ML algorithms more *autonomous* by learning *online* about the structure of the problem they are addressing. In the context of hyperparameter tuning in ML, sequential model-based optimization (SMBO, also known as Bayesian optimization; [82]) has gained interest recently [21, 126, 120]. Unlike gradient descent techniques, SMBO is an optimization paradigm that performs well in regimes where evaluations of the target function are costly. This is the case of the outer loop of hyperparameter tuning, since the computational cost of evaluating one set of hyperparameters is high. At each iteration, SMBO uses information such as previously evaluated points to build up a surrogate model of the function to be optimized, here the validation error, and then uses this model to pick the next point to be evaluated. The latter task is achieved by optimizing an auxiliary criterion that measures the interest of the evaluation of each point given the fitted model. Our vision is that in the near future, ML algorithms should be turn-key, with the hyperparameter tuning task completely automatized. We believe that SMBO is an appropriate framework to develop this automation, and we now present the contributions we made towards this goal, which correspond to Part I of this thesis.

First, Chapter 2 presents the SMBO framework, along with our contribution to the inner optimization of the auxiliary criterion. Auxiliary criteria, such as the well-known expected improvement criterion, are usually optimized by exhaustive search over the hyperparameter space, setting a grid or randomly sampling. We proposed a novel evolutionary optimization algorithm in [14] that outperforms exhaustive search on this subtask by exploiting the known structure of the expected improvement criterion. Second, to demonstrate that SMBO is up to the most challenging tasks in ML hyperparameter tuning, Chapter 4 presents tailored SMBO algorithms that tune the 30+ hyperparameters of DBNs, which we published in [21]. We first show – perhaps surprisingly – that random search outperforms manual tuning on these tasks, and that SMBO outperforms random search. Motivated by these positive results, we realized that a crucial advantage that human users still have over automatic tuning algorithms, when confronted to a new dataset, is their memory of past experiments over similar datasets, which gives them intuition on potentially good regions of the hyperparameter space. In Chapter 5, we present a yet unpublished framework named SCoT – for surrogate collaborative tuning – to perform model-based optimization of hyperparameters of learning algorithms *across datasets*. SCoT can, for instance, *(i)* take advantage of simultaneously tuning the same algorithm on several datasets and *(ii)* use the information gained from training the same algorithm on different datasets in the past. The main methodological difficulty arises when comparing the results of the considered learning algorithm on different datasets; traditional validation errors cannot be used directly, since two datasets can yield errors at arbitrarily different scales. The target of the algorithm, that is, the objective function of the overall optimization procedure, is thus ill-defined. SCoT solves this by relying on a surrogate ranking algorithm to design its target at each iteration. We present applications to AdaBoost, a popular classification algorithm. SCoT is also interesting from a pure methodological point of view, since it is a novel and generic SMBO algorithm that adapts its target on the fly.

Now let us come back to MCMC algorithms. The tuning of the proposal in MH can be compared to hyperparameter tuning in ML. Since the seminal paper [69], *adaptive* MCMC methods that learn their proposal distribution on the fly are an active research topic [9, 8]. Adaptive Metropolis (AM; [69]), for instance, aims at identifying the unknown covariance structure of the target distribution along the run of an MH algorithm with multivariate Gaussian proposal. The rationale behind this approach is based on scaling results which suggest that the chain correlation is minimized when the covariance matrix used in the proposal distribution matches, up to a constant that depends on the dimension, the covariance matrix of the target, for a large class of unimodal target distributions with independent marginals [111, 112]. AM thus progressively adapts, using a stochastic approximation scheme, the covariance of its proposal distribution to the estimated covariance of the target. AM is an example of MCMC algorithm with automatic hyperparameter tuning that takes into account what it learns about its task to adapt its sampling strategy online. This is similar in spirit to SMBO-based hyperparameter tuning in ML, although the goals and details are different. Part II of this thesis is devoted to adaptive MCMC methodology and applications.

Initially, our interest for adaptive MCMC was motivated by our involvement in the data analysis of the Pierre Auger experiment (henceforth denoted as Auger), a giant cosmic ray observatory that we present in Chapter 6. Cosmic rays are charged particles that travel through the universe at very high energies. When one of these particles hits our atmosphere, it generates a cascade of particles, among which *muons*, that strike the surface of Earth on several square kilometers. Auger has a $3\,000\,\mathrm{km}^2$ wide array of detectors gathering data from these cascades. The objective of the data analysis is to infer the parameters of the original incoming particles. For instance, the knowledge of the composition of the cosmic rays would have important implications in astrophysics, as reviewed in Chapter 6. We are thus interested in deriving a complete large-scale generative model of Auger to perform Bayesian inference on the parameters of each cascade. In Chapter 7, we present the bottom part of this model, which we first presented in [17], corresponding to the signal that muons generate in a single detecting unit. In the second half of Chapter 7, we describe an empirical Bayes procedure that only requires this bottom part of the model to yield results on Auger data.

Adaptive MCMC algorithms like AM are particularly suitable for inference in such intricate and high-dimensional models. However, the signal model presented in Chapter 7 is an additive model made of several components, one for each muon, which is invariant to permutations of these components. This invariance makes MCMC inference prone to *label switching*: the posterior distribution of all component parameters has exponentially many redundant modes, one for each permutation of the components, thus rendering the marginals of a well-mixing MCMC algorithm useless in practice. Furthermore, the use of adaptive proposal strategies is made impossible by the unidentifiability of the components. In Chapter 8, we review previous methods that address the label switching problem. None of these methods is simultaneously provably accurate, user-independent, and suitable for adaptive inference, which inspired us to design AMOR [14], an adaptive Metropolis algorithm with online relabeling that shares all three properties. We present AMOR here in Chapter 9. We then benchmark AMOR on a simple toy example to underline its behavior compared to previous approaches, on Bayesian inference in Gaussian mixture models, and on simulations from our Auger model described in Chapter 7. The first example is taken from a recently submitted paper [13], the second appeared in [14], and the third in [16].

Chapter 10 is devoted to the convergence proof of the AMOR algorithm that we also submitted in [13]. One of the technical challenges is that AMOR is adaptive both in its target and its proposal mechanism. Our convergence results are the first theoretical analysis of an online relabeling algorithm to our knowledge. The developed theoretical framework is appealing because *(i)* it generalizes previous approaches, *(ii)* it provides an explicit characterization of the posterior, *(iii)* it paves the way towards future work on the asymptotic behavior of relabeling algorithms and convergence of the relabeled samples, and *(iv)* it states an elegant and clear relation between relabeling and vector quantization techniques that was only subsumed in previous approaches.

Finally, in Chapter 11, we come back to links that are sketched in the main text, and make closing remarks on the future of some notions that are mentioned or developed in this thesis.

This thesis is pluridisciplinary, and we hope it will have readers with different backgrounds. Whereas it is primarily intended for readers with a statistics background, we tried to keep the general parts accessible and self-contained, provided chapters are read in the presentation order. Some parts, however, were too specific to maintain this objective, like the proofs in Chapter 10. Finally, notations are introduced along the main text to ease the reading, and we summarize the most frequent ones in Appendix A.

Part I

On the automatic tuning of machine learning algorithms

Chapter 2

Sequential model-based optimization

Contents

In this chapter, we review the paradigm of sequential model-based optimization and contribute to the inner loop of sequential GP-based optimization. This contribution is joint work with my advisor Balázs Kégl and was published in [15].

2.1 Introduction

There are numerous important global optimization problems in which the single evaluation of the target fitness function is very costly. Parameter optimization of large complex systems often requires running expensive simulations or carrying out real experiments that can take a long time. Hyperparameter optimization in machine learning is another example: evaluating one set of hyperparameters requires the full training that can take hours or days on today's large databases.

A natural way to deal with such problems is to replace the fitness function by a cheap-to-evaluate estimator, and optimize this surrogate model to propose a small number of points where the expensive fitness function is evaluated in an iterative active learning setup. This paradigm is called sequential model-based optimization (SMBO), Bayesian optimization, or surrogate-based optimization. Gaussian processes (GPs) provide an elegant way to model the fitness and to deal with the exploration-exploitation trade-off in a principled way. The paradigm of global optimization based on GPs dates back to the 1970s [97]. We start with some initial training points spread over the input space, evaluate the fitness function f at those points, and repeat the following steps:

1. Choose the next point to evaluate x^* by optimizing a cheap sampling criterion that measures some merit of an additional evaluation at any point of the input space,
2. Evaluate f at x^*.
3. Add $\big(x^*, f(x^*)\big)$ to the training set.

GP regression intervenes in the sampling criterion evaluation which involves the GP posterior over the fitness function given the training set.

The design of efficient sampling criteria (the so-called *merit functions*) is a hot research topic. Recent advances in this domain include the conditional entropy of the minimizer dealing with noisy evaluations (see [129] and the related [72]), the multi-armed bandit criterion [122] to derive regret bounds and the marginalized expected improvement and expected improvement per second of [120]. Our contributions do not depend on the choice of the criterion; we concentrate however throughout this thesis on the classical and commonly used *expected improvement* (EI) criterion of J. Mockus (see [82] for a recent extensive review). EI measures the expected amount by which we can improve the best fitness value obtained so far by going to a new point. Such criteria are usually highly multimodal, so optimization is typically done by a grid search or a Latin hypercube sampling approach that requires a large number of evaluations of the sampling criterion. This is a major drawback of these methods especially when the evaluation involves Monte Carlo sampling from the GP [129], so these methods are used mostly to optimize “expensive-to-evaluate” functions when the computational time to evaluate the functions justifies the time to be spent on proposing the next evaluation point.

In this chapter, we improve on the computational bottleneck of these methods by replacing the exhaustive evaluation of the surrogate and merit functions by a cross-entropy-based mixture method. The main idea is to replace the grid search by an adaptive evolutionary algorithm that iteratively samples in regions of higher merit value. The search distribution will be a mixture to take advantage of prior knowledge on the shape of the merit functions like EI. We have two contributions: a well-formulated mixture cross-entropy method and its application to model-based optimization. Experiments indicate that 1) we outperform the classical single-Gaussian cross-entropy method when the fitness function is highly multimodal, and 2) we outperform standard exhaustive search in GP-based surrogate optimization.

The outline of the chapter is as follows. In Section 2.2 we detail the SMBO paradigm and instantiate its inputs by decribing GPs and the EI criterion. Then in Section 2.3 we formally derive a mixture cross-entropy optimization method and propose an initialization procedure using triangulation of the training data. Finally in Section 2.4 we benchmark our mixture algorithm as a generic global optimization method, and compare it experimentally to exhaustive search on particular EI optimization problems.

2.2 Sequential model-based optimization

2.2.1 The paradigm

When tuning a learning algorithm on today's large databases, obtaining a validation error for a given set of hyperparameters requires a training phase that can typically take hours or days. Sequential model-based optimization (SMBO) is a surrogate optimization method suitable for such expensive-to-evaluate target functions. SMBO works by replacing the target function $f(x)$ by a cheaper-to-evaluate surrogate model, and iteratively 1) tune the model and 2) optimize an auxiliary criterion function $S(x)$ that measures the interest of asking the target function value at a new point x. The optimization paradigm is presented in Figure 2.1. The ingredients that the user has to provide are a model $\mathcal{M}$ and a criterion S, for which we now present two common choices that we used in our approach.

```
SMBO$(f, \mathcal{M}_0, T, S)$

  $\mathcal{O} \leftarrow \emptyset$
  For $t \leftarrow 1$ **to** $T$
      $x^* \leftarrow \operatorname{argmax}_x S(x, \mathcal{M}_{t-1})$
      Evaluate $f(x^*)$  ▷ *Expensive step*
      $\mathcal{O} \leftarrow \mathcal{O} \cup (x^*, f(x^*))$
      Fit a new model $\mathcal{M}_t$ to $\mathcal{O}$
  **return** $\mathcal{O}$
```

Figure 2.1: The pseudo-code of generic sequential model-based optimization. f is the function to be optimized, $\mathcal{M}_0$ is an initial surrogate model, T is the number of steps, and S is the auxiliary criterion that measures the interest of asking the target function value at a new point x.

The GP (also known as *kriging*) is a popular choice for model $\mathcal{M}$ in Figure 2.1, mainly due to its capacity to elegantly handle the uncertainty about the unknown fitness function. Several auxiliary criteria (also called merit functions, denoted by S in Figure 2.1) were proposed to handle the exploration-exploitation trade-off in global optimization. The most well-known are the *probability of improvement* and the *expected improvement* (EI; [82]). More recently [129] proposed to use the conditional entropy of the global minimizer to improve on EI when the the evaluation of the fitness function is noisy. One of the most recent novelty in the field is [122]'s proposal of using multi-armed bandits based on a GP surrogate model. After recalling the basics of GPs in Section 2.2.2 (based on [108]), we describe the most well-known criterion of expected improvement [82] in Section 2.2.3. We carried out all our experiments using this criterion; note, however, that the proposed technique is applicable with any GP-based merit function.

2.2.2 Gaussian processes

GPs provide a convenient way to put priors over functions. Let k be a positive definite kernel on the input space $\mathbb{X}$. As an example, a widely-used kernel for regressing smooth functions $f : \mathbb{R}^d \to \mathbb{R}$ is the squared exponential kernel:

$$k(x, y) = a \exp\left(-\frac{1}{2}(x-y)^T \Lambda^{-1}(x-y)\right) \tag{2.2.1}$$

where $\Lambda = \mathrm{Diag}(\ell_1^2, \ldots, \ell_d^2)$ is a diagonal matrix containing the so-called lengthscale parameters of k. An isotropic squared exponential kernel is then defined by taking $\Lambda = \ell I_d$, where I_d is the identity matrix of size d. A large number of kernels for real function regression can be found in [108].

Under a $\mathrm{GP}(0, k)$ prior, the distribution of any vector of function values $\mathbf{f} = (f(x_1), \ldots, f(x_n))^T$ is a multivariate Gaussian $\mathbf{f} \sim \mathcal{N}(0, K)$, where the so-called Gram matrix K is defined through $K_{ij} := k(x_i, x_j)$.

The most useful property of the GP prior is that it is closed under sampling: given a prior $p(f) \sim \mathrm{GP}(0, k)$ over functions and a set of samples

$$\mathcal{O} := \left\{(x_i, f(x_i)); 1 \leq i \leq n\right\},$$

the posterior $p(f|\mathcal{O})$ is also a GP with mean and covariance functions

$$\begin{aligned} \widetilde{m}(x) &= k(x, \mathbf{x})K^{-1}\mathbf{f}, \\ \widetilde{k}(x, x') &= k(x, x') - k(x, \mathbf{x})K^{-1}k(\mathbf{x}, x'). \end{aligned}$$

It is then straightforward to make predictions about the function value at a test point x^* since, according to the posterior, $f(x^*)$ has distribution $\mathcal{N}(\widetilde{m}(x^*), \widetilde{k}(x^*, x^*))$. Observe that the posterior variance at training points is zero, since the observations are noiseless. Additive homoscedastic Gaussian noise can be handled easily [108], resulting in replacing the Gram matrix K by $K + \sigma^2 I_n$, where $\sigma^2 > 0$ is the noise variance and I_n the identity matrix of size n. Heteroscedastic output noises [64] and input noises [95] can also be handled. An example fitted GP with homoscedastic noise can be seen on the top panel of Figure 2.2.

Finally, let us add that similarly to other kernel machines, GPs have their own hyperparameters η that are the parameters of the kernel k. For example, for the squared-exponential kernel in (2.2.1), η is $(a, \ell_1, \dots, \ell_d)^T$. These hyperparameters are commonly chosen so as to maximize the marginal likelihood $p(\eta|\mathcal{O})$ [108] and we will always follow this heuristics in this thesis. This optimization task is what "fitting a GP" actually means, as in Step 6 of the algorithm in Figure 2.1. First, we observe that evaluating the marginal likelihood requires inverting the Gram matrix K, thus making the computational complexity of fitting a GP cubic in the size of the training set $\mathcal{O}$. Second, we note that it was recently suggested in [18] and [120] that a full Bayesian treatment is more robust in practice: place a prior over η and sample from the posterior of η to marginalize quantities when needed, as for example when computing expected improvements.

2.2.3 The expected improvement criterion

Assume we want to minimize an unknown fitness function f, already evaluated at n points

$$\mathcal{O} := \{(x_i, f(x_i)); 1 \leq i \leq n\} \ .$$

The goal of EI is to find the next point x_{n+1} where the expected improvement over the currently best minimum $m_n := \min_i f(x_i)$ is the highest. As described in Section 2.2.2, fitting a GP on $\mathcal{O}$ yields, at every test point x^*, a mean $\widetilde{m}(x^*)$ and a standard deviation $\widetilde{\sigma}(x^*) = \widetilde{k}(x^*, x^*)^{1/2}$. The EI merit function is then defined by

$$\mathrm{EI}(x) := \mathbb{E}\big(\max(m_n - f(x), 0) | \mathcal{F}_n\big) \ ,$$

where $\mathcal{F}_n$ is the σ-algebra generated by the previous fitness evaluations in $\mathcal{O}$. An easy computation yields

$$\mathrm{EI}(x) = \widetilde{\sigma}(x)\big(u\Phi(u) + \phi(u)\big) \ , \tag{2.2.2}$$

where $u = (m_n - \widetilde{m}(x))/\widetilde{\sigma}(x)$, and Φ and ϕ denote the cdf and pdf of the $\mathcal{N}(0,1)$ distribution, respectively. This alternative definition is easier to understand: EI represents a trade-off between regions where the mean function is close to or better than the best value obtained so far and regions where the uncertainty is high. Notice that the EI merit function is always nonnegative and zero at every training point. It is generally smooth since it inherits the smoothness of the chosen kernel (which is in practice often at least once differentiable). The EI merit function is also likely to be highly multimodal, especially as the number of training points increases. Figure 2.2 illustrates an example of EI function. Our contribution in this chapter is an optimization algorithm which exploits this prior knowledge on the shape of the EI merit function to optimize it efficiently.

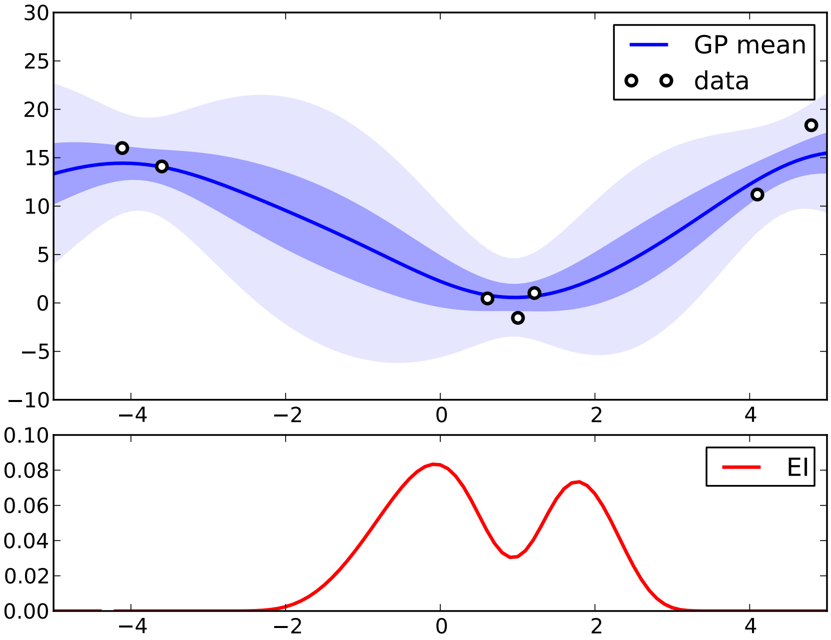


Figure 2.2: On the top panel, an example output of GP regression with a one-dimensional squared exponential kernel and additive homoscedastic, Gaussian noise is shown. The set $\mathcal{O}$ of previous evaluations of the target f is depicted as circles, the fitted posterior GP mean $\widetilde{m}$ is the thick blue line, while the dark and light blue shaded areas respectively depict one- and three-sigma error bars $\widetilde{m} \pm z\,\widetilde{k}(x,x)^{1/2}$, $x \in [-5,5]$, $z = 1, 3$. Shaded areas do not collapse at points in $\mathcal{O}$, indicating a nonzero noise level σ. If the top panel describes the prior on f, then the bottom panel shows the expected improvement criterion corresponding to minimizing f. Note that EI here is bimodal.

2.3 A mixture cross-entropy algorithm

Optimizing a GP-based merit function such as EI (2.2.2) is itself a difficult problem. Due to its multimodality, the most common technique is grid search either on a full grid or, especially in higher dimensions, using a Latin hypercube sampling. In any case, the criterion itself has to be evaluated a lot of times in each of the outer iterations of the global optimization loop. This can be slow even if the evaluation is analytical, let alone the case when the evaluation itself requires a Monte Carlo (henceforth MC) simulation from the GP [129]. For this reason, GP-based global optimization is often "marketed" as a technique for optimizing expensive functions, where the high computational complexity of evaluating the original fitness function f justifies the work invested in predicting the next evaluation point. In this section we describe an approach that can improve the computational complexity of the surrogate optimization, bringing GP-based global optimization closer to the family of generic global optimizers. Note that although we instantiate our contribution with the EI criterion in SMBO, it is a general optimization algorithm for highly multimodal functions.

Our approach uses importance sampling to adapt the search grid to the optimization problem. This is done by means of the *cross-entropy method* (CEM; see [117] for a detailed review) that we describe in Section 2.3.1. We make use of the fact that the multimodality of the merit functions suggests to model them with mixture distributions. Our main contribution is found in Section 2.3.2: we show how to use mixture distributions in CEM. Section 2.3.3 describes a specific initialization routine adapted to GP-based merit functions.

2.3.1 The cross-entropy method

The cross-entropy method is an adaptive importance sampling technique originally designed for integration on rare events. It proved to apply quite naturally to optimization. CEM provides a mathematical justification to evolutionary sampling methods, rigorously introducing the selection step in the estimation procedure. Consider computing

$$I := \mathbb{P}_u(A) = \int \mathbb{1}_A(x)\, g(x;u)\, dx \tag{2.3.1}$$

where the expectation is taken with respect to a pdf $g(x;u)$ belonging to some parametric family $\mathcal{G}$ indexed by u, and A is $\mathbb{P}_u$-rare. If one knows how to sample from $g(x;u)$, a crude Monte Carlo estimate of (2.3.1) is computable. But as A is rare for $\mathbb{P}_u$, few of the sampled points will happen to fall in A, so it is preferable to use importance sampling to reduce the variance of the MC estimator by sampling more points in the region of interest A. Importance sampling in this case consists in writing

$$I = \int \mathbb{1}_A(x)\, \frac{g(x;u)}{q(x)}\, q(x)\, dx \approx \frac{1}{N}\sum_{i=1}^{N} \mathbb{1}_A(x_i)\, \frac{g(x_i;u)}{q(x_i)} \tag{2.3.2}$$

for some distribution q chosen for easy sampling, whose support contains the support of $g(.;u)$ and $x_1, ..., x_N \sim q$ i.i.d. [110].

There is a theoretical answer to the question of the optimal choice of q, since if one takes $q = \tilde{q} \propto \mathbb{1}_A(.)g(.;u)$, the variance of the MC estimator will be 0. Of course, this is of no practical use since, to normalize q, the integral of interest I is needed, but one can still try to approximate $\tilde{q}$ in some sense. In particular, minimizing over $g(.;v) \in \mathcal{G}$ the Kullback-Leibler divergence between $\tilde{q}$ and $g(.;v)$ is equivalent to solve

$$\max_v \int \mathbb{1}_A(x)\, g(x;u)\, \log g(x;v)\, dx\ , \tag{2.3.3}$$

or, taking the empirical counterpart of (2.3.3) with eventually a new importance sampling step

$$\max_v \sum_{i=1}^{N} \mathbb{1}_A(x_i)\, \frac{g(x_i;u)}{g(x_i;w)}\, \log g(x_i;v) \tag{2.3.4}$$

where the x_i's are drawn i.i.d. according to $g(.;w)$.

Let us now turn this remark into an evolutionary optimization algorithm adapted to our original problem. Let us denote by S the criterion to maximize over $\mathbb{X}$. The key idea is to think of estimating probabilities of level sets, that is, integrals of the form $\mathbb{P}_u(S(X) \geq \gamma)$. Using the CEM principle to approximate the optimal importance distribution $\mathbb{1}_{(S(.)\geq\gamma)}g(.;u)$, the importance sampling paradigm will help us to sample points in $(S(X) \geq \gamma)$. Iteratively repeating the procedure while cleverly adapting γ to keep enough samples in the region of interest should lead us to sample from close to the optima of S.

Since we do not care about the actual estimate of the integral, we can get rid of the importance weights in (2.3.4) and iteratively optimize our choice of the importance distribution $g(.;v)$ to estimate $\mathbb{P}_{v_{t-1}}(S(X) \geq \gamma_t)$. The core algorithm finally proposed by the authors of [117] is given in Figure 2.3.

```
CEM for Optimization($S, N, \rho, d$)

1  Initialize $v_0$, set $t \leftarrow 1$
2  Sample $x_1, ..., x_N \sim g(.;v_{t-1})$ i.i.d.
3  Order $S(x_1), ..., S(x_N)$ decreasingly
4  Take $\gamma_t$ to be the $(1-\rho)$-quantile
   of the ordered performances
5  Solve
       $v_t := \arg\max_v \sum_{i=1}^{N} \mathbb{1}_{(S(x_i)\geq\gamma_t)} \log g(x_i;v)$        (2.3.5)
6  if $t \geq d$ and $\gamma_t = \gamma_{t-1} = ... = \gamma_{t-d}$ then stop
7  else set $t := t+1$ and go back to step 2
```

Figure 2.3: The CEM algorithm: the goal is to iteratively sample in regions of higher criterion value.

Note that taking $\mathcal{G}$ to be the family of Gaussians in CEM leads to the *estimation of multivariate normal algorithm* (EMNA; see [89] for a review on estimation of distribution algorithms and their applications), a popular evolutionary algorithm used in neural network training.

2.3.2 Introducing mixtures into the CEM

It is shown in [117] that (2.3.5) is analytically solvable when $\mathcal{G}$ is an exponential family. It turns out that there is a certain class of more generic distributions which would intuitively allow for a better fit of disconnected areas ($S \geq \gamma$), performing better exploration of multimodal landscapes by clustering the data: the mixture distributions. Their simplest form is a weighted sum of distributions belonging to parametric family $\Phi = (\varphi(.,v))_v$, i.e. $g(.;\mathbf{v}) := \sum_{d=1}^{D} \alpha_d \varphi(.;v_d)$ where $\sum_d \alpha_d = 1$. In the following subsection, we demonstrate that they also lead to analytical update formulae, whenever Φ is an exponential family.

The problem in its integral form is $\max_v \ell(\alpha, v)$, where $\ell(\alpha, v)$ denotes

$$\int \mathbb{1}_{(S(x)\geq\gamma)} \log \left(\sum_{d=1}^{D} \alpha_d \varphi(x; v_d) \right) g(x; v^{(t-1)})\, dx \; .$$

Let us define the posterior probability of x belonging to the dth cluster by $\rho_d(x; \alpha, v) \propto \alpha_d \varphi(x; \mu_d, \Sigma_d)$, and consider

$$L^t(\alpha, v) = \int \sum_{d=1}^{D} \mathbb{1}_{(S(x)\geq\gamma)}\, \rho_d(x; \alpha^{(t)}, v^{(t)}) \log \left(\alpha_d \varphi(x; v_d) \right) g(x; v^{(t-1)})\, dx \; .$$

By concavity of the log, we have

$$L^t(\alpha, v) - L^t(\alpha^{(t-1)}, v^{(t-1)}) \leq \ell(\alpha, v) - \ell(\alpha^{(t-1)}, v^{(t-1)}) \; ,$$

so any increase in L^t would mean a bigger-or-equal increase in ℓ. At the same time, maximization of $L^t(\alpha, v)$ leads to a closed formula whenever φ belongs to an exponential family, e.g. in the Gaussian case, writing

$$\omega^t_{d,\gamma}(x) = \rho_d(x; \alpha^t, \mu^t, \Sigma^t)\, \mathbb{1}_{(S(x)\geq\gamma)} \; ,$$

we derive

$$\begin{aligned}
\alpha_d^{t+1} &= \int \omega^t_{d,\gamma}(x)\, g(x; v^{(t)})\, dx \; , \\
\mu_d^{t+1} &= \frac{1}{\alpha_d^{t+1}} \int \omega^t_{d,\gamma}(x)\, x\, g(x; v^{(t)})\, dx \; , \\
\Sigma_d^{t+1} &= \frac{1}{\alpha_d^{t+1}} \int \omega^t_{d,\gamma}(x)\, (x - \mu_d^{t+1})(x - \mu_d^{t+1})^T g(x; v^{(t)})\, dx \; ,
\end{aligned}$$

whose empirical versions can be directly used as updates in Line 5 of Figure 2.3. Note that this new algorithm is very similar to population Monte Carlo schemes for integration [110].

2.3.3 Initialization via triangulation

CEM with mixtures fits the shape of the merit function, but it does not specify how to initialize the mixture components, a crucial step in practice. Recall that merit functions are zero at every training point and nonnegative everywhere. Their

Table 2.1: The three benchmark functions.

Function	Expression	Bound
sphere	$\sum_{i=1}^{d} x_i^2$	600
Rastrigin	$10d + \sum_{i=1}^{d} \left(x_i^2 - 10\cos(2\pi x_i)\right)$	5
Ackley	$20 + e - 20\exp\left(-0.2\sqrt{\frac{1}{d}\sum_{i=1}^{d} x_i^2}\right) -$ $\exp\left(\frac{1}{d}\sum_{i=1}^{d}\cos(2\pi x_i)\right)$	10

local maxima are *in between* the training points, so we should try to initialize the mixture components into these areas. The main idea is to triangulate the set of training points, and look at the modes in the interior of the simplices. We propose to initialize a component mean at the center of mass of every simplex and its covariance to $s^2 I$, where s is the distance from the center of mass to the nearest corner.

We chose to use the Delaunay triangulation[1] because it keeps the simplices as compact as possible [104], in the sense that the interior of the circumsphere of any simplex contains no training point. In the Euclidean plane, this is equivalent to saying that the minimum angle of Delaunay's triangles is maximum over all triangulations: every triangle is then *as equilateral as possible*. However, efficient implementations of Delaunay's triangulation for large datasets exist only for dimensions up to 6 [75]. In small dimensions, we could afford to build an initial training set containing the 2^d corners of a hypercubic search domain C, whereas in cases where $d \geq 6$, we replaced the complete set of corners by a small number of uniformly sampled corners.

2.4 Experiments

We present two sets of experiments. Since the mixture CE method is a novel proposal, we first benchmark it on three common test functions taken from the optimization community (Table 2.1). We then compare the mixture algorithm with Delaunay initialization against grid search on single steps of EI optimization with different training set sizes in two and ten dimensions.

2.4.1 Benchmarking the mixture CE algorithm

In this section we experimentally compare EMNA [89] to our mixture CE algorithm, using mixtures of Gaussians. EMNA is a popular evolutionary algorithm based on the on-line fit of a Gaussian surrogate model, and our algorithm can thus be seen as a generalization of EMNA, allowing to launch several EMNAs in different loci of the search space and adapt to the fitness landscape while favoring the best components. A mixture version of EMNA called *estimation of mixture of distributions algorithm*

[1] When $d \geq 3$, triangulation is to be understood as *simplexification*.

(EMDA) was already mentioned in [89]. Although our algorithm can be seen as an instance of EMDA, it is derived differently and it has the advantage of theoretically justifying the selection step by a ghost integration goal which is at the core of the CE method.

We started the optimization with a relatively large number of mixture components (10 for dimension 10, 20 for dimension 50), and gradually killed them when their mixing proportions went below a certain threshold (10^{-5} in our experiments). We used two different killing strategies. In the first strategy (red curves in Figure 2.4.1), we simply continued the procedure with the remaining components without replacing the killed ones. In the second strategy (green curves in Figure 2.4.1), we added a new component at the old component with highest mixing proportion, and assigned the new one a large initial variance to favor exploration around the current detected modes. We performed seven independent runs for each of the three algorithms. We plot the mean fitness obtained at the mean of the component with the highest mixing proportion versus the number of function evaluations. Shaded areas represent one standard deviation.

EMNA schemes are prone to degeneracy of the covariance matrix. To avoid it, we followed the softening advice of the authors of [117], taking at each time step $t = 1, 2, ...$ the new covariance matrix to be a weighted sum of the old covariance matrix and the selected sample covariance matrix, the weight of the latter being $\beta_t = \beta - \beta\,(1 - 1/t)^q$ with $\beta = 0.8$ and $q = 5$. This softening makes the degeneracy appear polynomially with the time rather than exponentially.

We chose common benchmark test functions in the continuous optimization community. We tried to reproduce the conditions of [89], initializing means uniformly over the initial ranges $[-B, B]^d$ where B is the specified bound in Table 2.1, taking $N = 2000$ points at each iteration and selecting the best $\mu = 1000$ points to compute the updates. We initialized all variances to 1. The three columns of Figure 2.4.1 depict our results on the Sphere, Rastrigin, and Ackley functions, respectively, the latter two being highly multimodal. The two rows correspond to dimensions $d = 10$ and $d = 50$. All functions are normalized to present a unique minimum at the origin.

Fitness graphs and spatial and eigenvalue-based diagnoses suggest that EMNA – with the degeneracy-avoiding update – finds the optimal mode after a reasonable number of function evaluations. Our mixture CE method seems to reach more quickly the best mode with either killing strategy: it automatically selects the best area according to its global model of the surface by focusing on the best components. We observed that after this pre-selection phase, the component means quickly concentrated on the best mode. After this step, the behavior was of course very similar to EMNA.

Let us insist on the fact that we used the same N and μ for EMNA and the mixture CEM, so updating a mixture costs the same price in function evaluations as updating a single distribution. That is why we think our algorithm can be considered as an automatic way of tuning the initialization of EMNA. Looking back to our original problem, this is exactly what we need in the context of EI optimization. Indeed, the EI landscape is possibly multimodal, and the mixture approach will allow us to visit the different modes, progressively select the best one, and finally sample only in the interesting area. As both killing strategies performed equally well, we will use the first – no replacement – in what follows, for the sake of simplicity.

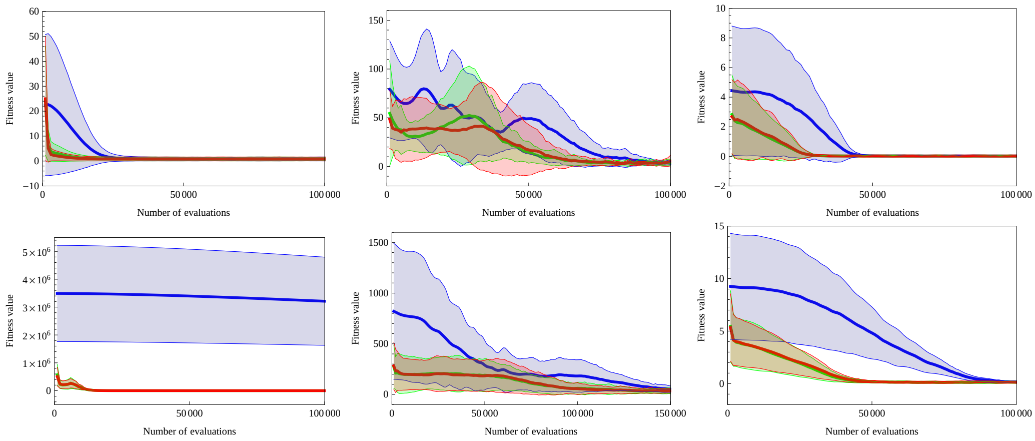


Figure 2.4: Empirical comparison of EMNA (blue) with two different killing strategies of our mixture CE algorithm (red and green, see the text for details) on the three benchmark functions (sphere, Rastrigin and Ackley, from left to right) in dimensions 10 (top) and 50 (bottom). Thick lines represent the mean fitness values of the components with highest mixing proportion, while shaded areas represent one-σ error bars.

2.4.2 A comparison on single EI steps: the setup

To verify experimentally whether the mixture method is more efficient than a grid search on the EI optimization problem, we propose the following setup. We considered the domain $C := [-5,5]^d$ with $d = 2, 10$, and we started by uniformly sampling $n' = 5, 20, 40$ points that we take as our training set, along with the (full or sampled) domain corners. The different values of n' represent different epochs in the final algorithm: as n' grows, we steer from an exploration phase to more advanced (exploitation) stage.

For each n', we optimized the EI criterion using grids of different resolutions. Then we ran our mixture CE method with an identical budget, meaning that it was only allowed to perform as many EI function evaluations as the number of points in the grid. For example, a 2D grid with a step size of 0.5 contains length$(-5 : 0.5 : 5)^2 =$ 441 points, so the red point corresponding to 0.5 in Figure 2.4.2 is the value obtained by the mixture algorithm after 441 EI function evaluations. The algorithms were run several times (3 times for each grid of stepsize r, with a shift uniformly distributed in $[0, r]$, and 5 times for each budget); the mean values obtained are plotted in thick lines, with shaded areas representing one-σ error bars.

The grid search becomes a real bottleneck in higher dimensions. The most popular solution is to replace the grid search by a Latin hypercube search, where the budget is a parameter. Figure 2.4.2 depicts a comparison of Latin hypercube search with our mixture search (in which we replaced the full set of corners by a subsample of 10), as detailed in Section 2.3.3.

The underlying fitness functions were the sphere, Rastrigin and Ackley functions, respectively (Table 2.1). The covariance function used in the experiments was an isotropic squared exponential with homoscedastic noise, for which the hyperparameters where tuned by maximizing the marginal likelihood of the GP [108].

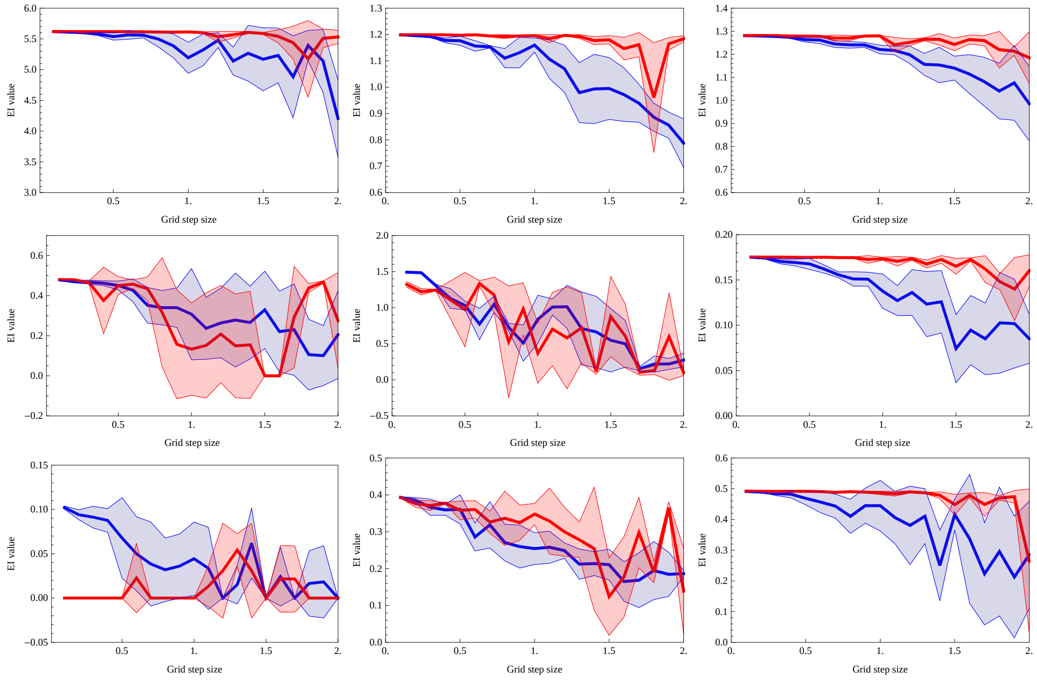


Figure 2.5: Empirical comparison of grid search (blue) and mixture search (red) on the three benchmark functions (sphere, Rastrigin and Ackley, from left to right) in dimension $d = 2$ with different training set sizes $n = 4 + n' = 9, 24, 44$ (from top to bottom). Thick lines represent the mean of the best EI values obtained, while shaded areas represent one σ error bars.

2.4.3 A comparison on single steps: comments

The mixture search outperforms the grid on all test functions in 2D for small training set size, and shows outstanding robustness to budget reduction for every training set size on the most difficult task (Ackley's function). On the sphere and Rastrigin's

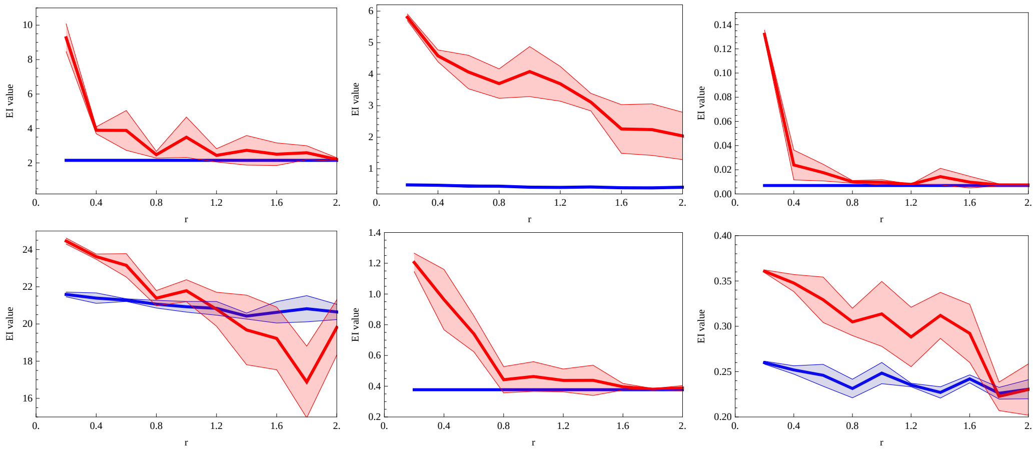


Figure 2.6: Empirical comparison of grid (blue) and mixture search (red) on the three benchmark functions (sphere, Rastrigin and Ackley, from left to right) in dimension $d = 10$ with training set sizes $n = 10 + n' = 17, 25$, from top to bottom. The x-axis r value corresponds to a budget of $(1 + 10/r)^2$. Thick lines represent the mean of the best EI values obtained, while shaded areas represent one σ error bars.

functions on medium and big size training sets, the mixture method remains competitive with the grid without outperforming it, while our method performs poorly on the sphere function with small training set.

Several empirical diagnoses can be invoked. The sphere function is the easiest to interpolate, and very quickly, the EI landscape consists only of a thin peak near the best value obtained so far. For Ackley's function, even after 40 iterations, the structure of the function was not explored enough, the support of the EI function is thus broader and allows for better estimation by the cross-entropy algorithm. Indeed, the elite sample of selected points is more representative of the EI function than in a case where almost all the samples fall in a very low fitness area of $C := [-5, 5]^d$.

In 10D, the comparison is even more favorable to the mixture method. It can clearly adapt to the landscape whereas the Latin hypercube sampling cannot. The Latin hypercube, and thus a grid, must contain a lot more points than the tested budgets to hope to find a decent mode. It is interesting to note that the only function that behaves differently is the sphere, for which Latin hypercube sampling seems more efficient for very low budget. But the implied EI values are quite high and do not give evidence for a too peaky EI landscape as in 2D. We think the problem might be the number of points given to the mixture algorithm: there must be an optimal trade-off between the number of iterations and the number of points for a given budget. This is similar to CMA-ES [71], another efficient search heuristics based on the evolutionary update of a single Gaussian. Notice that we forced in our tests the mixture search to perform at least five iterations, whatever the budget was (by decreasing the number of points per iteration).

2.5 Conclusion

We derived a new adaptive search method based on mixtures to solve the problem of merit maximization in GP-based global optimization. We gave it a theoretical justification through the link with the CEM, for which general convergence results are still in progress. Our method was experimentally shown to compare favorably with grid search in 2D with noticeable robustness to budget reduction, and to globally outperform Latin hypercube sampling in 10D. We believe that the proposed method can be particularly useful in GP-based global optimization when the merit function is not analytical so it needs Monte Carlo sampling from the GP.

Although the basic setup of the mixture CE method provides theoretical justifications if not guarantees, we had to make several heuristic algorithmic choices when implementing the practical method (the number of sample points from the mixture distribution, th e number of components, etc.). The triangulation initialization procedure is definitly a step that may be improved, especially in higher dimensions where the curse of dimensionality must be addressed. We could obviously accelerate the method by not throwing away the CE mixture components from one EI iteration to another, since it is likely that the EI surface would change significantly only around the newly added test point. This would eliminate the need of a heuristic initialization procedure in each iteration, however, it would probably make the procedure more sensitive to the birth/death policy of the mixture components.

An interesting direction to explore would be to further relate the method to Monte Carlo Markov chain (MCMC) integration. On the one hand, the optimized surrogate GP covariance could lead to an adaptive initialization of the CE mixture components, similarly to adaptive Metropolis-Hastings with Gaussian proposals (see [69] and Chapter 9 of this thesis). On the other hand, the birth/death policy could be governed by a principled procedure based on Reversible Jump MCMC techniques [68].

Chapter 3

A review on hyperparameter tuning

Contents

In this chapter, we review existing approaches to hyperparameter tuning in machine learning (ML) and related fields as an introduction to our contributions of Chapters 4 and 5. [77, Chapter 1] presents a good review of where other communities than ML stood in 2009. We adapt and update this review here.

3.1 Introduction

Most algorithms have high-level parameters that need to be chosen carefully to yield good performance. For instance, kernel-based classification and regression algorithms, such as support vector machines or Gaussian processes, typically require the tuning of their kernel parameters. Such user-dependent input parameters of an algorithm will be henceforth denoted as *hyperparameters*, and the process of choosing them will be called *hyperparameter tuning*. Outside of machine learning, it is sometimes called *algorithm configuration*.

Recent breakthroughs in classification have often come from better configuration of existing methods rather than the implementation of a new paradigm [103, 42, 44]. At the same time, deep architectures yield state-of-the-art results in classification, but are hard to set up for a non-expert: typical deep belief networks have tens of hyperparameters. Besides limiting the practical impact of new algorithmic discoveries, the hyperparameter tuning problem also makes it difficult to compare competing approaches based on empirical results obtained by different studies.

Eliminating the need for manual tuning was considered a major issue already in the 1990s in a lot of related fields: machine learning, planning, evolutionary optimization, SAT solving, etc. However, relatively few algorithmic solutions were proposed in spite of their potential practical importance and wide cross-disciplinary impact. In machine learning, for example, progressively refined grids or Latin hypercubes with occasional manual intervention remained standard.

For the formal description of the problem, let $\mathcal{A}$ be an algorithm. Let $\mathbb{H}$ be the set of all possible hyperparameter vectors for $\mathcal{A}$. $\mathbb{H}$ is often a product of continuous intervals and discrete sets. The methods presented here work by optimizing an objective function $f_{\mathcal{A}} : \mathbb{H} \to \mathbb{R}$ that takes a hyperparameter vector $x \in \mathbb{H}$ as input and outputs a measure of quality of x. In machine learning, this objective function is typically a validation or a cross-validation error[1]. When tuning an optimization algorithm, the objective function can be, for instance, the number of iterations needed to reach a fixed level set of a target function. We group here existing approaches in two big families according to whether they use a model of the objective function $f_{\mathcal{A}}$. This classification is only made for presentation purposes, and some algorithms could arguably be put in both groups.

3.2 Model-free tuning

This section reviews automatic hyperparameter tuning procedures that do not use a model of the objective function $f_{\mathcal{A}}$. These applications arose independently in different domains.

3.2.1 Common optimization heuristics

Be it to tune a machine learning, planning, or numerical optimization algorithm, once defined the application-specific objective function $f_{\mathcal{A}}$, common optimization heuristics can be applied to optimize it. Hill-climbing was used as early as [67].

[1] To the non-statistician reader: consider, for instance, a classification algorithm $\mathcal{A}$ that learns to assign a binary label to each point in $\mathbb{R}^d$, given a labeled dataset. The dataset is usually partitioned in three smaller sets; the validation error is the rate of points that are assigned the wrong label by algorithm $\mathcal{A}$ in the *validation dataset*, after all parameters of $\mathcal{A}$ have been fixed by minimizing an empirical error measure on the *train dataset*. After the user has found hyperparameters for $\mathcal{A}$ with a small enough validation error, he finally benchmarks $\mathcal{A}$ on the *test dataset*.

Later, more advanced heuristics were transposed and applied to hyperparameter tuning, such as the Nelder-Mead algorithm [98] and a variable neighborhood search with an acceptance step inspired by simulated annealing in [127]. Estimation of distribution algorithms (EDA; see [89] and Section 2.3.1 of this thesis) were also used recently [99], as well as ad hoc genetic algorithms in scheduling tasks [125].

3.2.2 Sophisticated grid searches

Sequentially and automatically refined grid searches were used for hyperparameter tuning in numerical optimization [10]. Precisely, it is proposed to alternately

1. perform a grid search,
2. automatically find and zoom on a promising region of $\mathbb{H}$.

Although the description of the algorithm in [10] uses a model of the objective, the method does not really rely on it and could be used with the original objective function, and we thus classify this algorithm as model-free. A similar approach, replacing exhaustive grids by *fractional experimental designs* – subsampled grids, such as Latin hypercubes –, was also proposed as a generic hyperparameter tuning method [2].

3.2.3 Racing

Traditionally, experimental design is associated to statistical statements on the value of $f_{\mathcal{A}}$ at the evaluated points of $\mathbb{H}$. Let $G \subset \mathbb{H}$ be a finite set of n candidate hyperparameter vectors. A racing algorithm launches in parallel n instances of $\mathcal{A}$, one per candidate hyperparameter vector in the grid G. The race has the following elimination rule: regularly, statistical tests are performed to compare the n instances, and the instances that are poorly ranked are discarded, meaning that their evaluation stops, thus concentrating the effort on reducing the uncertainty on well-performing instances. The F-race [24], for example, uses a Friedman test with null hypothesis stating that all instances are performing equally well. If this hypothesis is rejected, pairwise mean-comparison tests are applied to eliminate every instance that performed significantly worse than the current best instance.

Racing algorithms can be placed somewhere between the experimental design approaches of Section 3.2.2 and the pure exploration bandits with finite number of arms of [32]. Pure exploration bandits are bandit algorithms that seek to identify the best out of a given number of arms in a given time budget, instead of aiming at the traditional exploration/exploitation trade-off.

3.3 Model-based tuning

Evaluation of hyperparameters is the bottleneck in hyperparameter tuning: in machine learning, obtaining a validation error can easily take hours or days on today's large datasets. This limits the use of evaluation-greedy approaches such as exhaustive grids or gradient approximations. Model-based optimization consists in fitting a cheap-to-evaluate model to the objective function using all available information, and use this model to propose the next set of hyperparameters to be tested, as described in Chapter 2. We group here techniques according to what model they fit to the objective function.

3.3.1 Linear regression

In [45], a gradient descent algorithm is used, but not on $f_{\mathcal{A}}$ to avoid estimating its gradient. At each iteration of the gradient descent algorithm, a linear model $\ell : \mathbb{H} \to \mathbb{R}$ is fitted to $f_{\mathcal{A}}$ based on the results of the evaluation of $\mathcal{A}$ on a fractional experimental design, and the gradient step is made using the gradient of ℓ.

3.3.2 Neural network regression

In [30], several instances of the same planning problem are tuned at the same time. Each is tuned with a (1+1)-CMA-ES (covariance matrix adaptation evolution strategies; [71]) combined to genetic proposals. A one-layer neural network is fitted, not directly to the objective function, but to a hidden function that takes as input the features of an instance and outputs the best hyperparameters for this instance. The neural network is alternately trained on the available data and used to give hints to each CMA-ES. In this work, several instances of the same algorithm are tuned simultaneously and share the same model, an idea that we will build on in Chapter 5.

3.3.3 Gaussian process regression

Gaussian processes (GPs; [108]) enjoy analytical properties that make them a model of choice for $f_{\mathcal{A}}$ in sequential model-based optimization (SMBO, see Algorithm 2.1), as described in detail in Section 2.2. A reference work on hyperparameter tuning with GPs is [77], where different variants of SMBO tuning strategies are proposed. Again, this is close to pure exploration bandit algorithms, this time also with continuous arm spaces.

3.3.4 Bagging of regression trees

Another model that is particularly well suited for modelling an objective function with discrete inputs is the random forest [29]. SMBO algorithms with GPs replaced by bagged regression trees are presented in [77]. We note that the prediction uncertainty of the model is estimated in this work at a given hyperparameter vector by taking the empirical variance of the aggregated base predictors. Having marginal prediction and uncertainty allows to use similar strategies as with GPs to select the next hyperparameters to evaluate, such as maximizing EI (see Section 2.2.3).

3.4 Where do the contributions of this thesis fit?

In Chapter 2, we improved on the most basic and commonly-used GP-based optimizer. In the rest of Part I, we concentrate on the problem of model-based hyperparameter tuning, concentrating on ML applications, though the methods are generic enough to be applied to other fields. In Chapter 4, we develop a GP-based approach and a novel tree-based Parzen estimator (TPE) method to tackle the difficult task of tuning the 30+ hyperparameters of 3-layer deep neural networks. In short, the TPE is a model-based tuning algorithm in which the model is a Parzen-like estimator that takes advantage of the treed structure of the hyperparameter space in this particular problem. Finally, in Chapter 5, we present a novel SMBO framework to perform collaborative tuning *across datasets*, that is, to tune an ML algorithm on several datasets simultaneously, while making use of the information gathered in the past when tuning the same algorithm on other datasets.

3.5 Conclusion

While hyperparameter tuning has been widely considered a major issue, dedicated works are relatively sparse and state-of-the-art results in machine learning often are the result of a combination of exhaustive search and manual intervention. However, hyperparameter tuning in ML has clearly regained interest in the last two or three years with essentially the development and/or application of model-based algorithms as shown by recent publications on methodology or benchmarking. The development of generic software for hyperparameter tuning of ML algorithms has also received attention recently, with, for instance, the “Hyperopt” software package[2] [22]. Additionally, two recent papers propose detailed experimental comparisons of the model-based approaches we reviewed here, mostly applied to hyperparameter tuning in ML. In [120], a “fully Bayesian” variant of a GP-based method (see Section 2.2.2 for details) is shown to compare favorably with existing methods, including our TPE mentioned in Section 3.4 and usual GP-based methods on three different tasks, one of them being a recent difficult classification task. However, the dimension of the considered problems is relatively small (from 2 to 9 hyperparameters). In [126], a racing algorithm as in Section 3.2.3, the random forest approach of Section 3.3.4 and the TPE algorithm of Section 3.4 are compared on the task of choosing both

[2] `https://github.com/jaberg/hyperopt`

the algorithm and its hyperparameters to tackle various benchmarks from the UCI repository[3].

When the evaluation of a hyperparameter vector is costly enough that it becomes the bottleneck of the whole process, model-based methods provide useful tools. Now, they have their limits too, and GPs, for example, typically do not scale well to high dimensions either. However, we will show an application to deep learning in Chapter 4 where model-based tuning with GPs and more than 30 hyperparameters still outperform manual search. Note also that combining trees and GPs to obtain non stationary processes that are piecewise GP is also possible [66].

Finally, we look forward to a grand unification of racing, pure exploration bandits and Gaussian process-based optimization, which would probably give more practical insight into these methods. Such a unification is in the air, with the growing interest for Bayesian bandits [83] and the appearance of bandit-inspired GP-based optimization methods like [122].

[3] http://archive.ics.uci.edu/ml/

Chapter 4

Self-tuning deep learning

Contents

In this chapter, we describe concrete applications of the SMBO paradigm of Chapter 2 to the task of tuning the hyperparameters of deep learning algorithms, which are state-of-the-art learning algorithms widely studied in the ML community. The content of this chapter is joint work with James Bergstra (by the time at Université de Montréal, now at Harvard), his advisor Yoshua Bengio (Université de Montréal), and my advisor Balázs Kégl. It was published in [21].

4.1 Introduction

Models such as deep belief networks (DBNs; [74, 20]), stacked denoising autoencoders [130], convolutional networks [90], as well as classifiers based on sophisticated feature extraction techniques have from ten to perhaps fifty hyperparameters, depending on how the experimenter chooses to parametrize the model, and how many hyperparameters the experimenter chooses to fix to a reasonable default. The difficulty of tuning these models makes published results difficult to reproduce and extend, and makes even the original investigation of such methods more of an art than a science.

Recent results such as [103], [42], and [44] demonstrate that the challenge of hyperparameter optimization in large multilayer models is a direct impediment to scientific progress. These works have advanced state of the art performance on image classification problems by more concerted hyperparameter optimization in simple algorithms, rather than by innovative modeling or machine learning strategies. It would be wrong to conclude from a result such as [103] that feature learning is useless. Instead, hyperparameter optimization should be regarded as a formal outer loop in the learning process. A learning algorithm, as *a functional from data to classifier* (taking classification problems as an example), includes a budgeting choice of how many CPU cycles are to be spent on hyperparameter exploration, and how many CPU cycles are to be spent evaluating each hyperparameter choice (i.e. by tuning the regular parameters). The results of [103] and [44] suggest that with current generation hardware such as large computer clusters and GPUs, the optimal allocation of CPU cycles includes more hyperparameter exploration than has been typical in the machine learning literature.

This chapter makes two contributions: *(i)* Random search is competitive with the manual optimization of DBNs in the reference work [88], and *(ii)* sequential model-based optimization outperforms both manual and random search.

For notions relative to sequential model-based optimization and the expected improvement criterion, we refer the reader to Chapter 2 of this thesis. Section 4.2 introduces two approaches to optimize the EI criterion on the particular hyperparameter space of DBNs. Section 4.3 describes the experimental details of DBN hyperparameter optimization, and shows the efficiency of random search. Section 4.4 shows the efficiency of sequential optimization on the two hardest datasets according to random search. We conclude the chapter with a discussion of our results and final remarks in Section 4.5 and Section 4.6, respectively.

4.2 Optimizing EI on the DBN hyperparameter space

We present here two approaches to optimize the EI criterion on the particular hyperparameter space of DBNs.

4.2.1 Building up on the mixture CEM

Let $\mathbb{H}$ be the hyperparameter space of DBNs, and $f(x)$ be the classification error made by a DBN with hyperparameters $x \in \mathbb{H}$ on a fixed validation dataset. We put a GP prior on f and derive the expected improvement criterion as explained in Section 2.2.3. We modify the mixture cross-entropy presented in Section 2.3.2 as follows: we keep the EDA approach on the discrete part of our input space (categorical and discrete hyperparameters), where we sample candidate points according to binomial distributions, while we use the Covariance Matrix Adaptation - Evolution

Strategy (CMA-ES, [71]) for the remaining part of our input space (continuous hyperparameters). CMA-ES is a state-of-the-art gradient-free evolutionary algorithm for optimization on continuous domains, which has been shown to outperform the Gaussian search EDA [71]. Notice that such a gradient-free approach allows non-differentiable kernels for the GP regression. We do not take on the use of mixtures of Section 2.3.2, but rather restart the local searches several times, starting from promising places. The use of triangulations suggested in Section 2.3.3 is prohibitive here, as our task often means working in more than 10 dimensions, thus we start each local search at the center of mass of a simplex with vertices randomly picked among the training points. In the end, this search heuristics is very similar to the mixture CEM introduced in Section 2.3.2. Furthermore, the choice of multiple restart CMA-ES for this project was also dictated by the involvement of multiple programmers and the availability of good production code with practical convergence diagnoses for CMA-ES in several languages[1].

Finally, we remark that not all hyperparameters are relevant for every DBN structure. For example, a DBN with only one hidden layer does not have parameters associated to a second or third layer. Thus it is not enough to place one GP over the entire space of hyperparameters. We chose to group the hyperparameters by common use in a tree-like fashion and place different independent GPs over each group. As an example, for DBNs, this means placing one GP over common hyperparameters, including categorical parameters that indicate what are the *conditional* groups to consider, three GPs on the parameters corresponding to each of the three layers, and a few 1-dimensional GPs over individual conditional hyperparameters, like ZCA[2] energy (see Table 4.1 for a list of the considered DBN hyperparameters).

4.2.2 A tree-structured Parzen estimator approach

Anticipating that our hyperparameter optimization tasks will mean high dimensions and relatively small fitness evaluation budgets, a regime in which GPs show their limits, we now turn to another modeling strategy and EI optimization scheme for the SMBO algorithm in Figure 2.1.

Denoting by $y = f(x)$ the target validation error obtained with hyperparameters $x \in \mathbb{H}$, the GP approach of Section 4.2.1 builds a model of $p(y|x, \mathcal{F}_n)$, based on observations

$$\mathcal{O} = \{(x_i, y_i),\ 1 \leq i \leq n\} \subset \mathbb{H} \times \mathbb{R}\ .$$

Inspired by Parzen estimators, we propose here to directly build, still given observations $\mathcal{O}$, two approximations: $\ell(x) \approx p(x|y < y^*, \mathcal{F}_n)$ and $g(x) \approx p(x|y \geq y^*, \mathcal{F}_n)$. Before we give details on the construction of ℓ and g in Section 4.2.3, let us explain here heuristically how this approach is related to EI maximization. Note that, if we

[1] http://www.lri.fr/~hansen/cmaesintro.html

[2] Zero-phase Component Analysis (ZCA) is a variant of Principal Component Analysis (PCA).

consider that improvement means that the observed y is smaller than y^*, then the definition of EI in Section 2.2.3 yields

$$\mathrm{EI}(x) = \int_{-\infty}^{y^*} (y^* - y)\, p(y|x)\, dy = \int_{-\infty}^{y^*} (y^* - y)\, \frac{p(x|y)p(y)}{p(x)}\, dy \ , \tag{4.2.1}$$

where we dropped the conditioning on $\mathcal{F}_n$ for the sake of clarity.

Let now $\gamma = p(y < y^*)$. Upon noting that

$$\begin{aligned} p(x) &= \int p(x|y)p(y)\, dy \\ &\approx p(x|y < y^*)\, p(y < y^*) + p(x|y \geq y^*)\, p(y \geq y^*) \\ &\approx \gamma\ell(x) + (1-\gamma)g(x) \ , \end{aligned}$$

and

$$\int_{-\infty}^{y^*} (y^* - y)\, p(x|y)p(y)\, dy \approx \gamma y^* \ell(x) - \ell(x) \int_{-\infty}^{y^*} y\, p(y)\, dy \ , \tag{4.2.2}$$

we obtain

$$\mathrm{EI}(x) \approx \frac{\gamma y^* \ell(x) - \ell(x) \int_{-\infty}^{y^*} y\, p(y)\, dy}{\gamma \ell(x) + (1-\gamma) g(x)} \propto \left(\gamma + \frac{g(x)}{\ell(x)} (1-\gamma) \right)^{-1} \ .$$

This last expression shows that, roughly, to maximize improvement we would like points x with high probability under $\ell(x)$ and low probability under $g(x)$. The tree-structured form of the Parzen estimates ℓ and g will make it easy to draw many candidates according to ℓ and evaluate them according to $g(x)/\ell(x)$. On each iteration, our TPE algorithm returns the candidate x^* with the greatest g/ℓ ratio.

4.2.3 Details of the Parzen Estimator

The models $\ell(x)$ and $g(x)$ are hierarchical processes involving discrete-valued and continuous-valued variables. We detail the construction of ℓ, the construction of g being similar. Given n observations $\mathcal{O} = \{(x_i, y_i),\ 1 \leq i \leq n\} \subset \mathbb{H} \times \mathbb{R}$, let

$$\mathcal{B}_\ell = \{x_i : y_i \leq y^*\} \ ,$$

where y^* is chosen so that

$$\gamma n \leq \sum_{i=1}^{n} \mathbb{1}_{y_i \leq y^*} < \gamma n + 1 \ ,$$

and $\gamma \in (0, 1]$ is to be fixed by the user. Each continuous hyperparameter is specified by a uniform distribution over some interval (a, b), or a Gaussian, or a log-uniform distribution, see Table 4.1. The TPE substitutes an equally-weighted mixture of that distribution with Gaussians centered at each of the $x_i \in \mathcal{B}_\ell$. The standard

Table 4.1: Distribution over DBN hyperparameters for random sampling. Distribution specifications of the form A or B such as pre-processing (and including the random seed) correspond to equally weighted mixtures. Symbol U means uniform, $\mathcal{N}$ means Gaussian-distributed, and $\log U$ means uniformly distributed in the log-domain. CD (also known as CD-1) stands for contrastive divergence, the algorithm used to initialize the layer parameters of the DBN.

Whole model		**Per-layer**	
Parameter	**Distribution**	**Parameter**	**Distribution**
pre-processing	raw or ZCA	n. hidden units	$\log U(128, 4096)$
ZCA energy	$U(.5, 1)$	W init.	$U(-a, a)$ or $\mathcal{N}(0, a^2)$
random seed	5 choices	a	algo A or B (see text)
learn. rate	$\log U(0.001, 10)$	algo A coef.	$U(.2, 2)$
anneal. start	$\log U(100, 10^4)$	CD epochs	$\log U(1, 10^4)$
ℓ_2-penalty	0 or $\log U(10^{-7}, 10^{-4})$	CD learn. rate	$\log U(10^{-4}, 1)$
n. layers	1 to 3	CD anneal. start	$\log U(10, 10^4)$
batch size	20 or 100	CD sample data	yes or no

deviation of each Gaussian was set to the greater of the distances to the left and right neighbor, but clipped to remain in a reasonable range. In the case of the uniform, the points a and b were considered to be potential neighbors. For discrete variables, supposing the prior was a vector of N probabilities p_i, the posterior vector elements were proportional to $Np_i + C_i$ where C_i counts the occurrences of choice i in $\mathcal{B}_\ell$. The log-uniform hyperparameters were treated as uniforms in the log domain.

4.3 Random search as the new baseline

One simple but recent step towards formalizing hyperparameter optimization is the use of random search [103]. [23] showed that random search was much more efficient than grid search for optimizing the parameters of one-layer neural network classifiers. In this section, we evaluate random search for DBN optimization, compared with the sequential grid-assisted manual search carried out in [88].

We chose the distributions listed in Table 4.1 to define the search over DBN configurations. The details of the datasets, the DBN model, and the greedy layer-wise training procedure based on contrastive divergence (CD; [34]) are provided in [88]. The distributions of Table 4.1 correspond to the search space of [88] except for the following differences:

1. we allowed for ZCA pre-processing [78],
2. we allowed for each layer to have a different size,
3. we allowed for each layer to have its own training parameters for CD,

4. we allowed for the possibility of treating the continuous-valued data as either as Bernoulli means (more theoretically correct) or Bernoulli samples (more typical) in the CD algorithm, and

5. we did not discretize the possible values of real-valued hyperparameters.

These changes expand the hyperparameter search problem, while maintaining the original hyperparameter search space as a subset of the expanded search space.

The results of this preliminary random search are in Figure 4.1. Perhaps surprisingly, the result of manual search can be reliably matched with 32 random trials for several datasets. The efficiency of random search in this setting is explored further in [23]. Where random search results match human performance, it is not clear from Figure 4.1 whether the reason is that it searched the original space as efficiently, or that it searched a larger space where good performance is easier to find. But the objection that random search is somehow cheating by searching a larger space is backward – the search space outlined in Table 4.1 is a natural description of the hyperparameter optimization problem, and the restrictions to that space by [88] were presumably made to simplify the search problem and make it tractable for grid-search assisted manual search. Critically, both methods train DBNs on the same datasets.

The results in Figure 4.1 indicate that hyperparameter optimization is harder for some datasets. For example, in the case of the "MNIST rotated background images" dataset (**MRBI**), random sampling appears to converge to a maximum relatively quickly (best models among experiments of 32 trials show little variance in performance), but this plateau is lower than what was found by manual search. In another dataset (**convex**), the random sampling procedure exceeds the performance of manual search, but is slow to converge to any sort of plateau. There is considerable variance in generalization when the best of 32 models is selected. This slow convergence indicates that better performance is probably available, but we need to search the configuration space more efficiently to find it. The remainder of this chapter explores sequential optimization strategies for hyperparameter optimization for these two datasets: **convex** and **MRBI**.

4.4 Benchmarking SMBO for hyperparameter tuning in DBNs

4.4.1 Validating surrogate modelling on the "Boston housing" dataset

We validated our GP approach of Section 4.2.1 by comparing with random sampling on the "Boston Housing" dataset, a regression task with 506 points made of 13 scaled input variables and a scalar regressed output. We trained a *multi-layer perceptron*

(MLP) with 10 hyperparameters, including learning rate, ℓ_1 and ℓ_2 penalties, size of hidden layer, number of iterations, whether a PCA pre-processing was to be applied, whose energy was the only conditional hyperparameter [25]. Our results are depicted in Figure 4.2(a). The first 30 iterations were made using random sampling, while from the 30th on, we differentiated the random samples from the GP approach trained on the updated history. The experiment was repeated 20 times. Although the number of points is particularly small compared to the dimensionality, the surrogate modelling approach finds noticeably better points than random, which supports the application of SMBO approaches to more ambitious tasks and datasets.

Applying the GP to the problem of optimizing DBN performance, we allowed 3 random restarts to the CMA+ES algorithm per proposal x^*, and up to 500 iterations of conjugate gradient method in fitting the length scales of the GP. The squared exponential kernel [108] was used for every node. The CMA-ES part of GPs dealt with boundaries using a penalty method, the binomial sampling part dealt with it by nature. The GP algorithm was initialized with 30 randomly sampled points in $\mathcal{H}$. After 200 trials, the prediction of a point x^* using this GP took around 150 seconds.

For the TPE-based algorithm, we chose $\gamma = 0.15$ and picked the best among 100 candidates drawn from $\ell(x)$ on each iteration as the proposal x^*. After 200 trials, the prediction of a point x^* using this TPE algorithm took around 10 seconds. TPE was allowed to grow past the initial bounds used with for random sampling in the course of optimization, whereas the GP and random search were restricted to stay within the initial bounds throughout the course of optimization. The TPE algorithm was also initialized with the same 30 randomly sampled points as were used to seed the GP.

4.4.2 Parallelizing sequential search

Both the GP and TPE approaches were actually run asynchronously in order to make use of multiple compute nodes and to avoid wasting time waiting for trial evaluations to complete. For the GP approach, the so-called *constant liar* approach was used: each time a candidate point x^* was proposed, a fake fitness evaluation equal to the mean of the y's within the training set $\mathcal{O}$ was assigned temporarily, until the evaluation completed and reported the actual loss $f(x^*)$. For the TPE approach, we simply ignored recently proposed points and relied on the stochasticity of draws from $\ell(x)$ to provide different candidates from one iteration to the next. The consequence of parallelization is that each proposal x^* is based on less feedback. This makes search less efficient, though faster in terms of wall time.

Runtime per trial was limited to 1 hour of GPU computation regardless of whether execution was on a GTX 285, 470, 480, or 580. The difference in speed between the slowest and fastest machine was roughly two-fold in theory, but the actual efficiency of computation depended also on the load of the machine and the configuration of the problem (the relative speed of the different cards is different in different hyperparameter configurations). With the parallel evaluation of up to five proposals from the GP and TPE algorithms, each experiment took about 24 hours of wall time using five GPUs.

4.5 Results and discussion

The performance of the trajectories constructed by each algorithm up to 200 steps are illustrated in Figure 4.3, and compared with random search and the manual search carried out in [88]. The generalization scores of the best models found using these algorithms and others are listed in Table 4.2(b). On the **convex** dataset (2-way classification), both algorithms converged to a validation score of 13% error. In generalization, TPE's best model had 14.1% error and GP's best had 16.7%. TPE's best was significantly better than both manual search (19%) and random search with 200 trials (17%). On the **MRBI** dataset (10-way classification), random search was the worst performer (50% error), the GP approach and manual search approximately tied (47% error), while the TPE algorithm found a new best result (44% error). The models found by the TPE algorithm in particular are better than previously found ones on both datasets. The GP and TPE algorithms were slightly less efficient than manual search: GP and TPE identified performance on par with manual search within 80 trials, the manual search of [88] used 82 trials for **convex** and 27 trials for **MRBI**.

There are several possible reasons for why the TPE approach outperformed the GP approach in these two datasets. Perhaps the direct modelling of $p(x|y)$ is more accurate than the $p(y|x)$ in the Gaussian process. Perhaps, conversely, the exploration induced by the TPE's lack of accuracy turned out to be a good heuristic for search. Perhaps the hyperparameters of the GP approach itself were not set to correctly trade off exploitation and exploration in the DBN configuration space. More empirical work is required to test these hypotheses. Additionally, after we published these results in [21], a GP-approach with fully Bayesian tuning (see Section 2.2.2) has been reported to outperform in its turn our TPE approach on tasks of smaller dimension [120]. Critically though, all our four SMBO runs matched or exceeded both random search and a careful human-guided search, which are currently the state of the art methods for hyperparameter optimization.

The GP and TPE algorithms work well in both of these settings, but there are certainly settings in which these algorithms, and in fact SMBO algorithms in general, would not be expected to do well. Sequential optimization algorithms work by leveraging structure in observed (x, y) pairs. It is possible for SMBO to be arbitrarily bad with a bad choice of $p(y|x)$. It is also possible to be slower than random sampling at finding a global optimum with an apparently good $p(y|x)$, if it extracts structure in $\mathbb{H}$ that leads only to a local optimum.

4.6 Conclusion

This chapter has introduced two sequential hyperparameter optimization algorithms, and shown them to meet or exceed human performance and the performance of a brute-force random search in two difficult hyperparameter optimization tasks involving DBNs. We have relaxed standard constraints (e.g. equal layer sizes at all layers) on the search space, and fall back on a more natural hyperparameter space of 32 variables (including both discrete and continuous variables) in which many variables are sometimes irrelevant, depending on the value of other parameters (e.g. the number of layers). In this 32-dimensional search problem, the TPE algorithm presented here has uncovered new best results on both of these datasets that are significantly better than what DBNs were previously believed to achieve. Moreover, the GP and TPE algorithms are practical: the optimization for each dataset was done in just 24 hours using five GPU processors. Although our results are only for DBNs, our methods are quite general, and extend naturally to any hyperparameter optimization problem in which the hyperparameters are drawn from a measurable set.

We hope that our work may spur researchers in the machine learning community to treat the hyperparameter optimization strategy as an interesting and important component of all learning algorithms. The question of "How well does a DBN do on the **convex** task?" is not a fully specified, empirically answerable question – different approaches to hyperparameter optimization will give different answers. Algorithmic approaches to hyperparameter optimization make machine learning results easier to disseminate, reproduce, and transfer to other domains. The specific algorithms we have presented here are also capable, at least in some cases, of finding better results than were previously known. Finally, powerful hyperparameter optimization algorithms broaden the horizon of models that can realistically be studied; researchers need not restrict themselves to systems of a few variables that can readily be tuned by hand.

The TPE algorithm presented in this work, as well as parallel evaluation infrastructure, is available as BSD-licensed free open-source software, which has been designed not only to reproduce the results in this work, but also to facilitate the application of these and similar algorithms to other hyperparameter optimization problems.[3]

[3] "Hyperopt" software package: `https://github.com/jaberg/hyperopt`

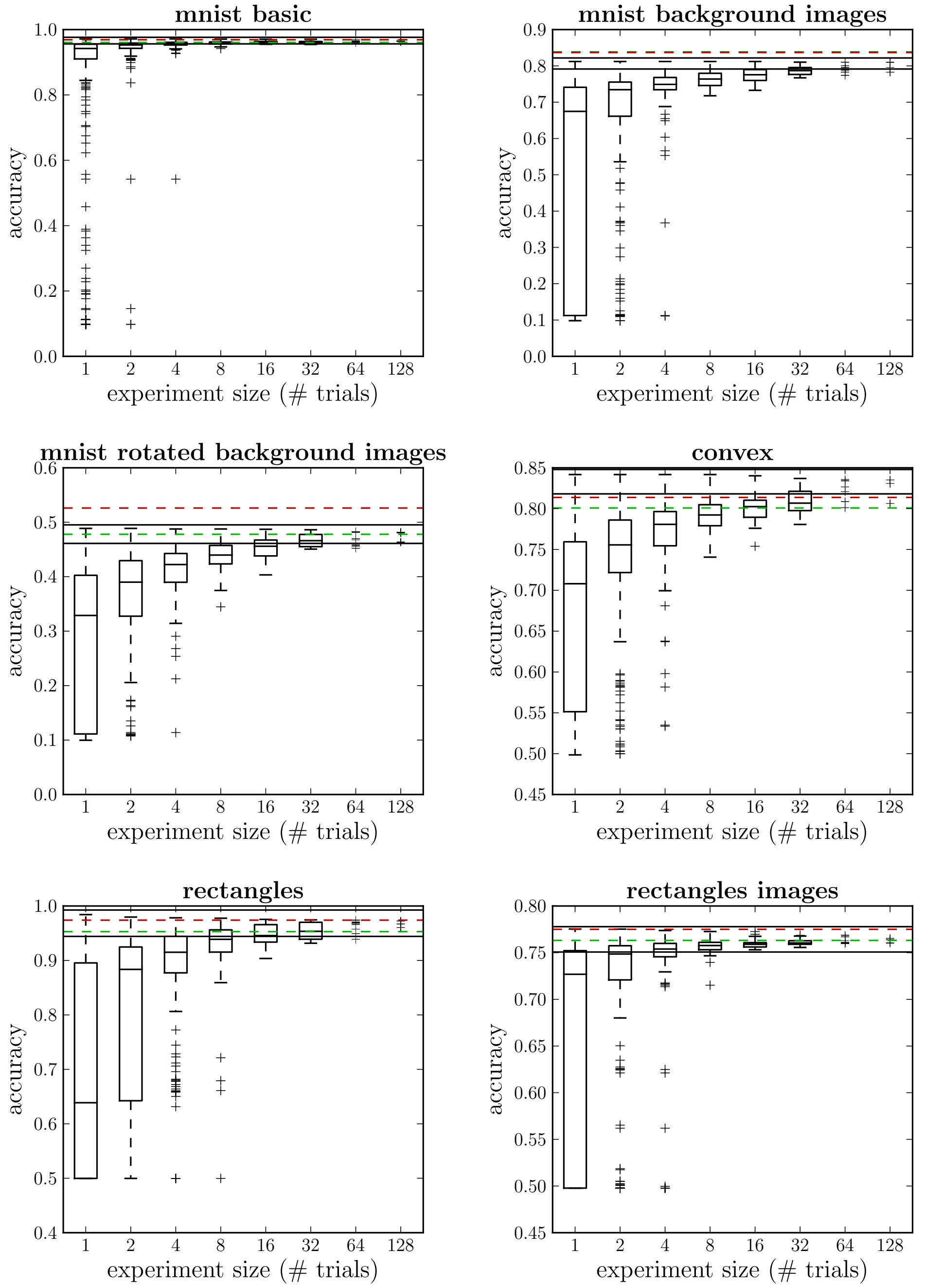
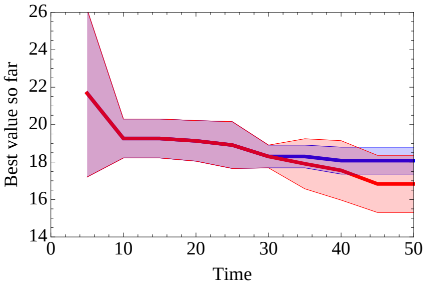


Figure 4.1: Deep Belief Network (DBN) performance according to random search. Random search is used to explore up to 32 hyperparameters (see Table 4.1). Results found using a grid-search-assisted manual search over a similar domain with an average 41 trials are given in green (1-layer DBN) and red (3-layer DBN). Each box-plot (for $N = 1, 2, 4, ...$) shows the distribution of test set performance when the best model among N random trials is selected. The datasets "convex" and "mnist rotated background images" are used for more thorough hyperparameter optimization.



(a) "Boston housing" results

	convex	**MRBI**
TPE	**14.13** ±0.30 %	**44.55** ±0.44%
GP	16.70 ± 0.32%	47.08 ± 0.44%
Manual	18.63 ± 0.34%	47.39 ± 0.44%
Random	18.97 ± 0.34 %	50.52 ± 0.44%

(b) **convex** and **MRBI** test errors

Figure 4.2: (a) After iteration 30, GP optimizing the MLP hyperparameters on the "Boston Housing" regression task. Thick lines depict the best minimum found so far every 5 iterations, against time; red is for the GP+EI approach of Section 4.2.1, blue is for random uniform search. Shaded areas indicate one-sigma error bars. (b) The test set classification error of the best model found by each search algorithm on each problem. Each search algorithm was allowed up to 200 trials. The manual searches used 82 trials for **convex** and 27 trials **MRBI**.

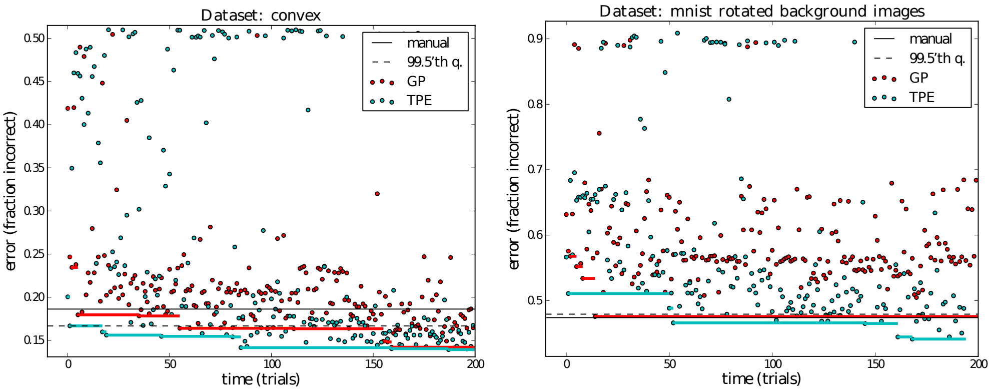


Figure 4.3: Efficiency of Gaussian Process-based (GP) and graphical model-based (TPE) sequential optimization algorithms on the task of optimizing the validation set performance of a DBN of up to three layers on the **convex** task (left) and the **MRBI** task (right). The dots are the sequences produced by each SMBO algorithm. The solid colored lines are the validation set accuracy of the best trial found before each point in time. Both the TPE and GP algorithms make significant advances from their random initial conditions, and substantially outperform the manual and random search methods. A 95% confidence interval about the best validation means on the **convex** task extends 0.018 above and below each point, and on the **MRBI** task extends 0.021 above and below each point. The solid black line is the test set accuracy obtained by domain experts using a combination of grid search and manual search [88]. The dashed line is the 99.5% quantile of validation performance found among trials sampled from our prior distribution (see Table 4.1), estimated from 457 and 361 random trials on the two datasets respectively.

Chapter 5

Surrogate collaborative tuning

Contents

In this chapter, we contribute a yet unpublished framework named SCoT – for surrogate collaborative tuning – to perform model-based optimization of hyperparameters of learning algorithms *across datasets*. In other words, SCoT presents the two novelties of 1) tuning a learning algorithm on *several* new datasets and 2) taking into account the information gained from training the same algorithm on similar datasets in the past. We present applications to AdaBoost, a popular classification algorithm. The content of this chapter is joint work with Matthias Brendel (by the beginning of our collaboration at Université Paris-Sud, now at Ferchau Engineering, Germany), Michèle Sebag (CNRS, Université Paris-Sud) and my advisor Balázs Kégl.

5.1 Introduction

We have seen in Chapter 5 a successful application of sequential model-based optimization to the tuning of hyperparameters of deep belief networks. What may still make experienced practitioners better at hyperparameter optimization is their ability to generalize *across* similar learning problems. For example, if somebody in the past successfully applied a classification algorithm $\mathcal{A}$ to the popular MNIST dataset with a given set of hyperparameters $x \in \mathbb{H}$, he or she would certainly use this set as a hint (or *prior*) to choose the hyperparameters of $\mathcal{A}$ when tuning $\mathcal{A}$ on a slightly noisy or rotated version of MNIST. Our main contribution is to propose a way to mimic this human behavior when performing an automatized, surrogate-based hyperparameter search.

As mentioned in Section 3.3.2, a related idea was recently proposed in [30], in which the authors tuned an evolutionary algorithm by learning a mapping from problems to hyperparameters using a neural network. This setting may work well if the problem descriptors determine the optimal hyperparameters with high certainty. In machine learning on diverse data sets, however, we cannot hope to come up with a set of easily measurable problem descriptors that can correctly predict the best hyperparameters. What we propose here instead is to build a model from past experience that can bias the search on a new problem towards regions in the hyperparameter space where the optimal hyperparameters are likely to be found. Surrogate-based methods suit well this setup. They also solve efficiently the exploration/exploitation dilemma. Furthermore, combining surrogate-based ranking and optimization techniques, we can define a novel Bayesian optimization method that can be used to collaboratively optimize quantitatively different but similarly behaving objective functions.

Imagine for now that we have picked up N_f features of datasets that allow us to consider each dataset as a point in $\mathbb{D} \subset \mathbb{R}^{N_f}$. To tackle the problem, we first build a *quality function* $f_\mathcal{A} : \mathbb{D} \times \mathbb{H} \to \mathbb{R}$ that takes a dataset $D \in \mathbb{D}$ and a set of hyperparameters $x \in \mathbb{H}$ as inputs, and outputs the quality $f_\mathcal{A}(D, x)$ of the result $\mathcal{A}(D, x)$ obtained by applying $\mathcal{A}$ on D using hyperparameters x. We then place a prior over $f_\mathcal{A}$ that incorporates knowledge from previous experiments. The Gaussian process (GP) prior is now a common choice in surrogate-based global optimization [82, 91]. We have mentioned in Chapter 2 that since GPs are closed under sampling, they provide a simple and elegant way of accumulating more and more knowledge about the surrogate function as the observations arrive. On the other hand, specifying the quality function $f_\mathcal{A}$ is more subtle. Typically, $f_\mathcal{A}$ could be a (cross-) validation error $f_\mathcal{A}(D, x) = R\big(\mathcal{A}(D, x)\big)$ as in Chapter 4. The problem with this choice is that, when applying the same algorithm $\mathcal{A}$ to different problems $D_1, \ldots, D_M \in \mathbb{D}$, the errors can differ significantly, so that the raw validation error is a poor choice for $f_\mathcal{A}$. At the same time, similar problems may share a model about *where* the error is minimized in the hyperparameter space. To deal with this issue, we need a quality function that encodes knowledge on the ranking of hyperparameters on individual problems, and can convey information such that:

$$\begin{gathered}\textit{if } f_\mathcal{A}(D_1, x_1) < f_\mathcal{A}(D_1, x_2) \textit{ and } D_2 \textit{ is similar to } D_1,\\ \textit{then probably } f_\mathcal{A}(D_2, x_1) < f_\mathcal{A}(D_2, x_2),\end{gathered}$$

even though the ranges of $R\big(\mathcal{A}(D_1, \cdot)\big)$ and $R\big(\mathcal{A}(D_2, \cdot)\big)$ are very different. We propose therefore to *rank* hyperparameters x_j on each problem D_i by the corresponding validation error $R\big(\mathcal{A}(D_i, x_j)\big)$, and take $f_{\mathcal{A}}(D_i, x_j)$ to be the surrogate model output by a surrogate-based *ranking* algorithm. This results in a novel and rather uncommon SMBO algorithm that completely redefines its training set at each time step, since new rankings yield a new model.

For a description of the SMBO paradigm, we refer the reader to Chapter 2 of this document. The rest of the chapter is organized as follows: Section 5.2 contains our methodological contributions, describing in detail the quality function $f_{\mathcal{A}}$ and the prior we place over it. Finally, we present in Section 5.3 an experimental case study of our algorithm on AdaBoost, where we compare our approach to several realistic human behaviors.

5.2 The quality function and its prior

To explain our motivation for designing the quality function and choosing a ranking-based surrogate method, we start by an example. Figures 5.1(a) and 5.1(b) show the two-dimensional error surfaces when running AdaBoost.MH of [118] on two benchmark sets (the data selection and the experimental setup will be described in detail in Section 5.3). The sets are similar in terms of some high-level features (such as the number of instances, number of classes, or number of attributes) and the shapes of the error surfaces are also similar, so it intuitively makes sense to try to model the rankings defined by these two error functions by a common surrogate function. The errors $R\big(\text{AdaBoost}(D, (m, T))\big)$, however, are quantitatively different, so that direct fitting of a common model would not make sense. To overcome this problem, we will use a ranking surrogate $f_{\text{AdaBoost}}\big(D, (m, T)\big)$ that only defines the relative ordering of error values.

5.2.1 A fictitious generative model

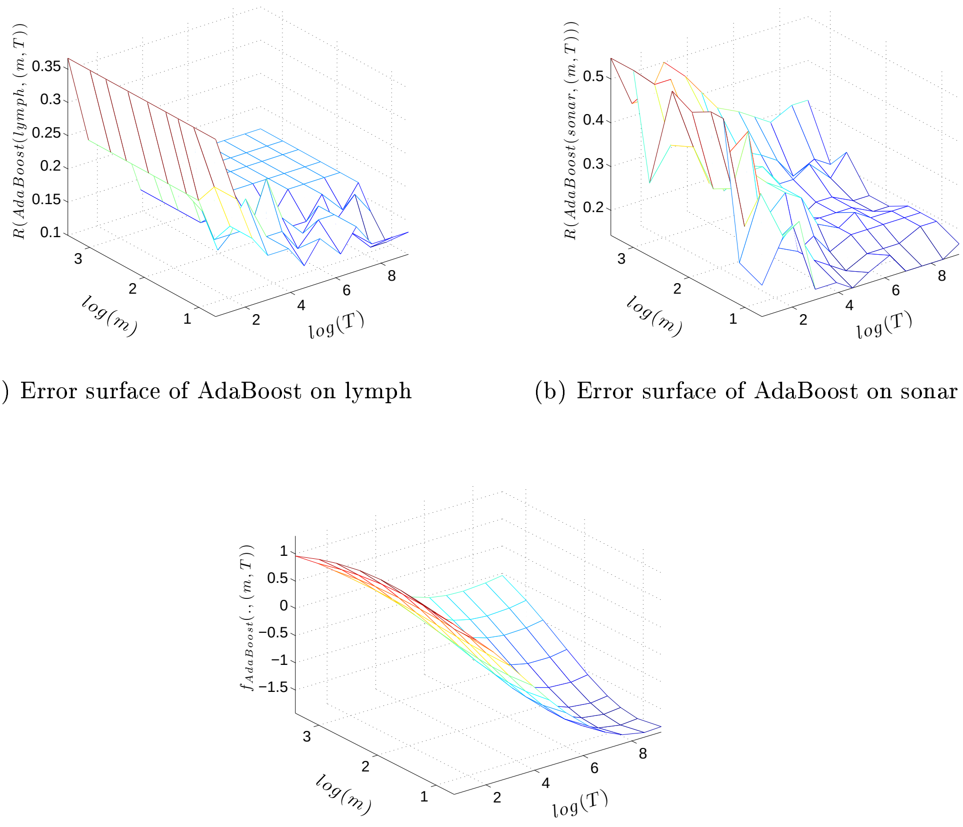
To formally define the hypothesis under which our approach makes sense, we can describe an imaginary generative model that produces error surfaces using a common underlying ranker. Our algorithm will then perform inference in this model in a Bayesian-like approach.

Let us fix a learning algorithm $\mathcal{A}$. Let $R\big(\mathcal{A}(D, x)\big)$ be a validation error of algorithm $\mathcal{A}$ applied with hyperparameters $x \in \mathbb{H}$ to problem $D \in \mathbb{D}$. For a fixed problem D, define a ranking $\prec_D$ over hyperparameters by

$$x_1 \prec_D x_2 \Leftrightarrow R\big(\mathcal{A}(D, x_1)\big) \leq R\big(\mathcal{A}(D, x_2)\big) \; . \tag{5.2.1}$$

Assume that there exists a smooth function $f^*_{\mathcal{A}} : \mathbb{D} \times \mathbb{H} \to \mathbb{R}$ that preserves rankings in the sense that for each problem D, $x_1 \prec_D x_2$ implies $f^*_{\mathcal{A}}(D, x_1) < f^*_{\mathcal{A}}(D, x_2)$. Now define an equivalence relation on functions by

$$f \sim g \Leftrightarrow \exists \sigma : \mathbb{R} \to \mathbb{R} \text{ smooth and strictly increasing s.t. } f = \sigma(g) \; .$$



(a) Error surface of AdaBoost on lymph

(b) Error surface of AdaBoost on sonar

(c) The common latent ranker

Figure 5.1: (a,b) Error surfaces on two similar datasets have similar shapes although the errors are quantitatively different in terms of scale. (c) The similar shapes can be captured by a latent ranker.

Then any member of the equivalence class $\mathcal{C}^*_{\mathcal{A}}$ of $f^*_{\mathcal{A}}$ is a smooth function that preserves rankings. When a problem D_i is drawn from the problem space $\mathbb{D}$, it comes with an arbitrary smooth and strictly monotonic function σ_i that takes the output of the ranking function $f^*_{\mathcal{A}}(D_i, x)$ as input, and outputs the error $R\big(\mathcal{A}(D_i, x)\big) = \sigma_i\big(f^*_{\mathcal{A}}(D_i, x)\big)$. In principle, we could draw data

$$\mathcal{O} = \Big(D_i, x_i, \sigma_i\big(f^*_{\mathcal{A}}(D_i, x_i)\big)\Big)_{i=1:N} \tag{5.2.2}$$

from an arbitrary generative distribution over $\mathbb{D} \times \mathbb{H}$. In reality, the problems cannot be arbitrarily generated, so the set $\mathbb{D}$ will be finite, and we will draw hyperparameter vectors $x_1, \ldots, x_N$ for each dataset.

Assuming that $f^*_{\mathcal{A}}(D, x)$ is smooth in x is quite natural, and most of the local search and surrogate optimization algorithms are designed based on this hypothesis, including the ones developed in Chapters 2 and 4. In addition, and this is our key assumption, we also suppose that $f^*_{\mathcal{A}}(D, x)$ is smooth in D, which means that similar problems produce similarly-looking error surfaces. How problem similarity is defined is an interesting and generally unanswered question. In Section 5.3 we propose a simple setup, but the algorithm described in the following sections is generic in the sense that it only assumes that there exists a positive semidefinite kernel representing problem similarity.

5.2.2 A deconvolution method

Our final algorithm, when confronted to choosing hyperparameters for algorithm $\mathcal{A}$ on a new problem D, will aim at minimizing $f^*_{\mathcal{A}}(D, \cdot)$. That is why we now start from data $\mathcal{O}$ in (5.2.2) and present a way to perform inference on $f^*_{\mathcal{A}}$. Surrogate-based learning algorithms such as SVM$^{\text{RANK}}$ [81] or the GP-based ranking algorithm of [40] build a function $\widehat{f}_{\mathcal{A}}$ that preserves the rankings they have been given while keeping $\widehat{f}_{\mathcal{A}}$ sufficiently smooth and flat. This cancels the influence of σ_i in the generation and we thus consider $\widehat{f}_{\mathcal{A}}$ as a noisy realization of a representer of $\mathcal{C}^*_{\mathcal{A}}$. We then perform GP regression [108] on this realization $\widehat{f}_{\mathcal{A}}$ and pick the next point to add to $\mathcal{O}$ by maximizing EI, as described in Chapter 2. Note that for the moment, the optimization of EI is restricted to the subspace $\{D\} \times \mathbb{H}$.

Iteratively repeating the process described above yields a first Bayesian optimization algorithm presented in Figure 5.2, named ST for Surrogate-based Tuning. What makes it a particularly unusual Bayesian optimization algorithm is that the regressed function is different at each time step: it is the targeted equivalence class $\mathcal{C}^*_{\mathcal{A}}$ that is fixed.

```
ST$\big(D, T, \mathcal{O} = (\mathcal{D}, \mathcal{H}, \mathcal{R}), \mathcal{A}, \mathcal{B}\big)$

for $t \leftarrow 0$ to $T - 1$
    Compute rankings $\mathcal{P}_t$ from $\mathcal{O}$ as in (5.2.1)
    $\widehat{\mathbf{f}}_t \leftarrow$ surrogate model built by $\mathcal{B}$ called on $(\mathcal{D}_t, \mathcal{H}_t)$ with rankings $\mathcal{P}_t$
    $M_{t-1} \leftarrow$ Posterior GP on $\widehat{f}_t$ knowing $\big((\mathcal{D}_t, \mathcal{H}_t), \widehat{\mathbf{f}}_t\big)$
    $x^* \leftarrow \operatorname{argmax}_{x \in \mathbb{H}} EI(D, x)$
    $R^* \leftarrow R(\mathcal{A}(D, x^*))$    ▷ Run learning algorithm
    $\mathcal{O} \leftarrow \mathcal{O} \cup (D, x^*, R^*)$
return $\mathcal{O}$
```

Figure 5.2: The pseudo-code of the surrogate-based tuning algorithm. Input $\mathcal{B}$ denotes a surrogate-based ranking algorithm. Input $\mathcal{O}$ summarizes results from past experiments with the same algorithm. See text for details.

5.2.3 Collaborative tuning

It is common that a user wants to apply his learning algorithm to several problems $D_1, \ldots, D_M$. Instead of repeatedly applying ST (Figure 5.2), our surrogate-based approach allows to tune all problems simultaneously by spending one iteration on each problem in turn, thus making use of all information gained so far *on all problems*. This gives birth to the SCoT algorithm, for surrogate-based collaborative tuning, presented in Figure 5.3.

```
SCoT$\big((D_1, \ldots, D_M), T, \mathcal{O} = (\mathcal{D}, \mathcal{H}, \mathcal{R}), \mathcal{A}, \mathcal{B}\big)$

1 for $t \leftarrow 0$ to $T - 1$
2     for $i \leftarrow 1$ to $M$
3         Compute rankings $\mathcal{P}_t$ from $\mathcal{O}$ as in (5.2.1)
4         $\widehat{\mathbf{f}}_t \leftarrow$ surrogate model built by $\mathcal{B}$ called on $(\mathcal{D}_t, \mathcal{H}_t)$ with rankings $\mathcal{P}_t$
5         $M_{t-1} \leftarrow$ Posterior GP on $\widehat{f}_t$ knowing $\big((\mathcal{D}_t, \mathcal{H}_t), \widehat{\mathbf{f}}_t\big)$
6         $x^* \leftarrow \mathrm{argmax}_{x \in \mathbb{H}}\ EI(D_i, x)$
7         $R^* \leftarrow R(\mathcal{A}(D_i, x^*))$   ▷ Run learning algorithm
8         $\mathcal{O} \leftarrow \mathcal{O} \cup (D_i, x^*, R^*)$
9 return $\mathcal{O}$
```

Figure 5.3: The pseudo-code of the surrogate-based collaborative tuning algorithm. Input $\mathcal{B}$ denotes a surrogate-based ranking algorithm. Input $\mathcal{O}$ summarizes results from past experiments with the same algorithm. See text for details.

5.2.4 On the choice of a surrogate-based ranking algorithm

The method of choice for Algorithm $\mathcal{B}$ in Figure 5.3 is the Gaussian Process-based ranking algorithm of [40] since it provides a way of simultaneously estimating the values of a hidden $\widehat{f}$ and tuning the GP hyperparameters in a double optimization for loop. This method, applied with a squared exponential kernel with automatic relevance determination (SE-ARD; [108]), would avoid the need for Step 5 of SCoT and provide an easier interpretation for $\widehat{f}$. However, both our implementation and the one available on the web proved to be too slow in the regime presented in Section 5.3 to be realistically incorporated in the for loop of SCoT. We thus chose to limit ourselves in Step 4 to an isotropic squared exponential kernel, and used the efficient optimization routines available for SVM$^{\text{RANK}}$ [81], while Step 5 is carried out with an SE-ARD kernel. Note that SVM$^{\text{RANK}}$ requires that all hyperparameter sets must be comparable when applied to the same problem, which is the case in our framework.

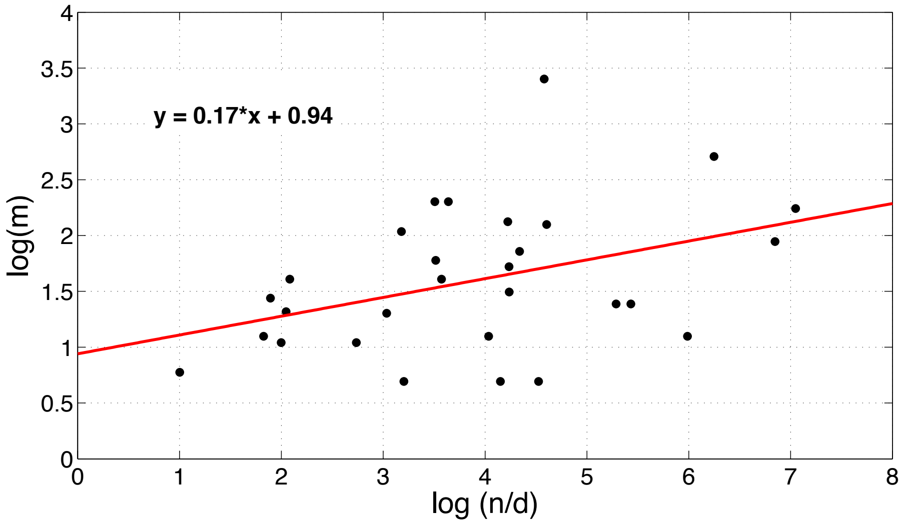


Figure 5.4: Optimal number of product terms versus the problem feature $\log(n/d)$.

5.3 A case study on AdaBoost

To test SCoT as a practical hyperparameter tuning algorithm, we now describe a setup that mimics an experienced AdaBoost user on multi-class classification problems. We used the implementation of AdaBoost.MH available at `multiboost.org` [19]. Let $D \in \mathbb{D}$ be a dataset. From now on, points in D are called *instances*, and the components of a point in D are called *attributes*. The term *feature* will be reserved to numerical descriptors of the whole classification problem, i.e. the components of D.

5.3.1 Setup

We downloaded 29 multi-class classification problems from Weka. We converted nominal attributes to numerical (binary) values using a one-hot encoding. We split each set 80-20% into training and validation sets. We ran AdaBoost.MH with decision products as weak learners [84], so that the two hyperparameters to tune were the number of iterations T and the number of product terms m. Out target measure $R\big(\textsc{AdaBoost}(D,(m,T))\big)$ was the classical multi-class 0-1 error. To simplify the algorithmic setup, we pre-computed all validation errors on a 12×9 grid with values

$$m \in \{2, 3, 4, 5, 7, 10, 15, 20, 30\}$$

and

$$T \in \{2, 5, 10, 20, 50, 100, 200, 500, 1000, 2000, 5000, 10000\}\ ,$$

giving us 108 trained models for each dataset. The inputs of the GP kernel were the logarithms of the hyperparameters, rescaled to belong to $[0, 1]$. Figure 5.1 shows two example error surfaces and a smooth latent ranker learnt by the GP.

We now describe the choice of the features describing each dataset. To embed the problems into a Euclidean space where a GP can be used with a classical squared exponential kernel, we first extracted three simple measures from each dataset, the number of training instances n, the number of classes K, and the number of attributes d, and defined three features K, $\log d$, and $\log(n/d)$ that we found indicative about the value of the best hyperparameters. For example, Figure 5.4 shows that $\log(n/d)$ is correlated to the optimal number of product terms m, which makes sense: the more instances we have compared to the number of attributes, the more complex the optimal classifier can be.

Table 5.1: Minimum value, first quartiles, medians, third quartiles and maximum value of the features of the problems.

feature	min	1st q.	med	3rd q.	max
K	2	2	5	10	26
$\log d$	1.61	2.3	2.94	4.01	5.49
$\log(n/d)$	1	3.03	4.04	4.57	7.05
ρ	0.14	0.5	0.67	0.75	0.86

The fourth feature ρ is called here PCA reduction rate, and was computed as follows: we extracted the first d' principal components that explained 95% of the variance of each dataset, and divided d' by the number of attributes d. Table 5.1 summarizes the statistics of the features, and Figure 5.5 visualizes the datasets in the feature space projected onto the first two principal components. The first two components explain 73.5% of the variance, which means that the four-dimensional distribution is not degenerate but not completely uniform either. Finally, the features were rescaled to belong to $[0, 1]$.

5.3.2 Experiments

We designed five experiments, each one mimicking a different user behavior. In all experiments, we used a 5-fold cross-validation over the 29 datasets. To avoid confusion with training and testing in each problem, we call *meta-train* and *meta-test*, respectively, the train and test sets made out of problems.

Global default This experiment mimics the tuning strategy of a non-expert in machine learning who chooses a unique hyperparameter vector for all problems. Although this sounds unreasonable, a typical Weka user often runs classification algorithms using the default hyperparameters. Here we assume that the designer of the classification algorithm is an expert, so he or she sets the

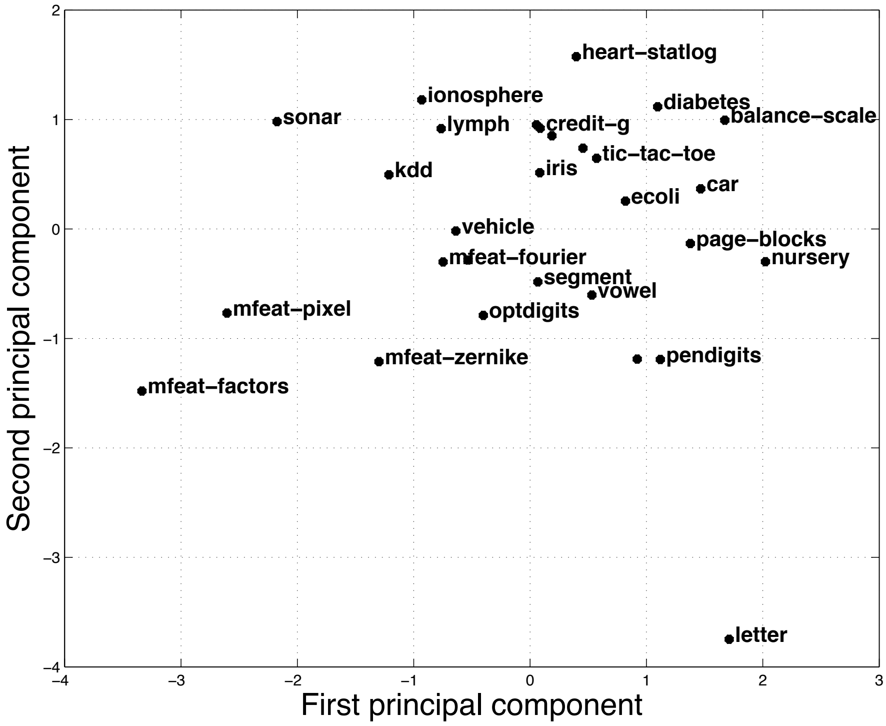


Figure 5.5: Projection of the problems onto the first two principal components of the feature space. For the sake of visibility some problem names are omitted.

default hyperparameters to those that perform the best on average. Formally, we select the hyperparameter vector that minimizes the average error over the problems in the meta-train sets.

Collaborative default This experiment mimics the tuning strategy of a more experienced user, who builds a model according to his or her experience with the meta-train set, but runs out of time before the conference deadline. She thus uses the surrogate model to predict one vector of hyperparameters for each problem, but does no tuning on the problems. In practice, this strategy is similar to a single iteration of the outer loop of SCoT (Figure 5.3) and consists in *(i)* taking the surrogate output by SVM$^{\text{RANK}}$, *(ii)* regressing it with a GP, and *(iii)* taking the hyperparameter with the best posterior mean on each meta-test problem.

Separate surrogate tuning This experiment mimics a user who has time for a sequential algorithm, but does not learn either from the meta-train set or from the knowledge acquired in tuning on other meta-test problems simultaneously. State-of-the-art surrogate tuning methods, such as the ones presented in Chapter 4, are of this kind. This experiment consists in running a different SMBO algorithm on each meta-test problem, thus using an independent two-dimensional GP for each meta-test problem. To avoid numerical problems, we start by four random iterations.

SCoT (surrogate collaborative tuning) This experiment mimics an expert user, who tries to learn a relation between features, hyperparameters, and quality from the meta-train problems he encountered in the past. Here we combine the static model and the surrogate technique, and we add collaborative tuning as described in Section 5.2 and Figure 5.3.

Random search This is the baseline experiment that samples points uniformly over the grid, without replacement. This is probably the most common strategy when exhaustive grid search is out of computational reach. [23] demonstrated that random search is competitive in hyperparameter tuning for classical multilayer perceptrons and deep belief networks, see also Chapter 4.

5.3.3 Results

Figure 5.6 shows the average meta-test error as a function of the number of iterations. The curves are (obviously) similar in the beginning and at the end of the experiment, but between step 20 and 50, the speedup of reaching a given error level can be more than two-fold wrt. separate tuning, and more than three-fold wrt. random search.

Comparing the methods in terms of average generalization error might however be questionable for the exact same reason that made us use a ranking-based surrogate: the classification datasets may not be commensurable in terms of simple, generalization error. Hence we computed an average rank score in the following manner: in each iteration and for each problem, the results of the different strategies were ranked with the so-called fractional ranking (also known as “1 2.5 2.5 4” ranking), where ties are rewarded by the average of their ordinal rankings. For each strategy, the ranking points of all meta-test problems were then averaged to get the final score. The average rank can then be computed for each method and plotted against the number of trials. Note that the average rank of a single method depends on the results of the others, which means that non-tuning methods (global and collaborative defaults) will get higher and higher scores while their quality values remain constant. Also, note that the lower the average rank score, the better is the method.

Figure 5.7 shows the results. The first observation is that the collaborative default achieves a better score than the global default, validating therefore our hypothesis that past experience can help to find better hyperparameters. Secondly, separate tuning beats random search, confirming the results of [21]. Finally, SCoT seems to robustly outperform all methods which means that combining surrogate optimization and collaborative tuning gets the best of both worlds. Note that as the number of iterations grow, all three tuning methods start to saturate the search space, so their average rank converges to the same value.

5.3.4 Computational issues

Since SCoT is meant as a prototype for real-life hyperparameter tuning, it is important to look at computational costs. The training of the surrogate models in separate tuning are negligible compared to running MultiBoost. The cost of training a model on the meta-train problems in the collaborative approaches is considerable: each fold contains 23 or 24 problems, each one associated with 108 hyperparameter vectors, which means that we need to train the latent ranker on 2500 points in total. In practice, it is realistic to learn, that is, to apply SVM$^{\text{RANK}}$ and to tune the parameters of the GP kernel, this model offline. It is also plausible to update this model regularly with new points the user has tried out. What is computationally too expensive is to completely re-train the model at each iteration of an SMBO algorithm, since this would not compete with exhaustive search if cost was raw computational time time (versus the number of runs of $\mathcal{A}$). Our strategy was therefore the following. In general once SVM$^{\text{RANK}}$ finds the values of the surrogate ranker $f_{\mathcal{A}}$, we optimize the hyperparameters of the GP using CMA-ES [71], a local search optimization algorithm. This optimization is run for 1000 iterations on points from the meta-train. Once we start tuning the hyperparameters of a meta-test problems, we add new points to the model one by one. Instead of re-optimizing the GP hyperparameters each time from scratch, we start from the last model and run CMA-ES only for 100 steps. In practice, this optimization of GP hyperparameters can disappear once the number of meta-train problems is large.

5.4 Conclusion

We presented SCoT, a surrogate-based optimization algorithm for hyperparameter tuning. It builds on previous approaches, adding a memory of past experiments and a collaborative tuning ability. Developing these two new points has led to the combination of surrogate-based optimization and ranking methods, resulting in a novel Bayesian optimization algorithm whose target is moving with the iteration number. We demonstrated SCoT in a case study on AdaBoost, where it outperformed a variety of common tuning strategies, including vanilla surrogate-based optimization. Its main limit is the computational cost of maintaining a Gaussian process model over a huge set of problem-hyperparameter pairs, which we by-passed by fully training the model only once in the beginning, then only updating it with shorter optimization routines to keep the overall time-scale realistic for future applications.

In order to eventually yield a (semi) automatic hyperparameter tuner, some methodological and theoretical efforts are still needed. On the methodological side, a comprehensive database of learning problems, progressively closing the gap between benchmarks problems and a real-world application, should be compiled. On the theoretical side, some efforts of feature construction are needed to define, for instance, compound parameters, more amenable to build the surrogate quality model. Such augmentations of the training set of SCoT will necessitate careful choices in the methods employed, maybe leading to lighten the computational burden with cheaper models, such as approximate GPs (see [108, Chapter 8], and [38]). Finally, we feel there is room for improvement in designing asynchronous strategies for the collaborative tuning of SCoT, which currently tunes problems in a synchronous way, spending one point on each problem in turn, while it should intuitively spend more evaluations on difficult problems.

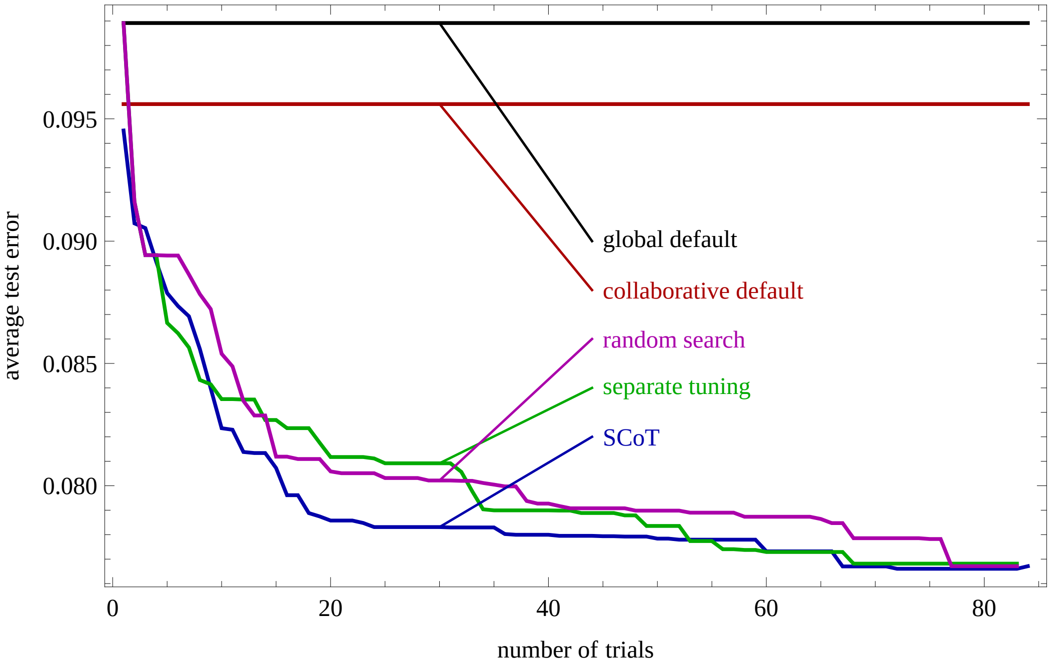


Figure 5.6: The average meta-test generalization error as a function of the number of trials. The curves are (obviously) similar in the beginning and at the end of the experiment, but in the middle of the experience, SCoT can reach a given average error twice as fast as separate tuning and three times as fast as random search.

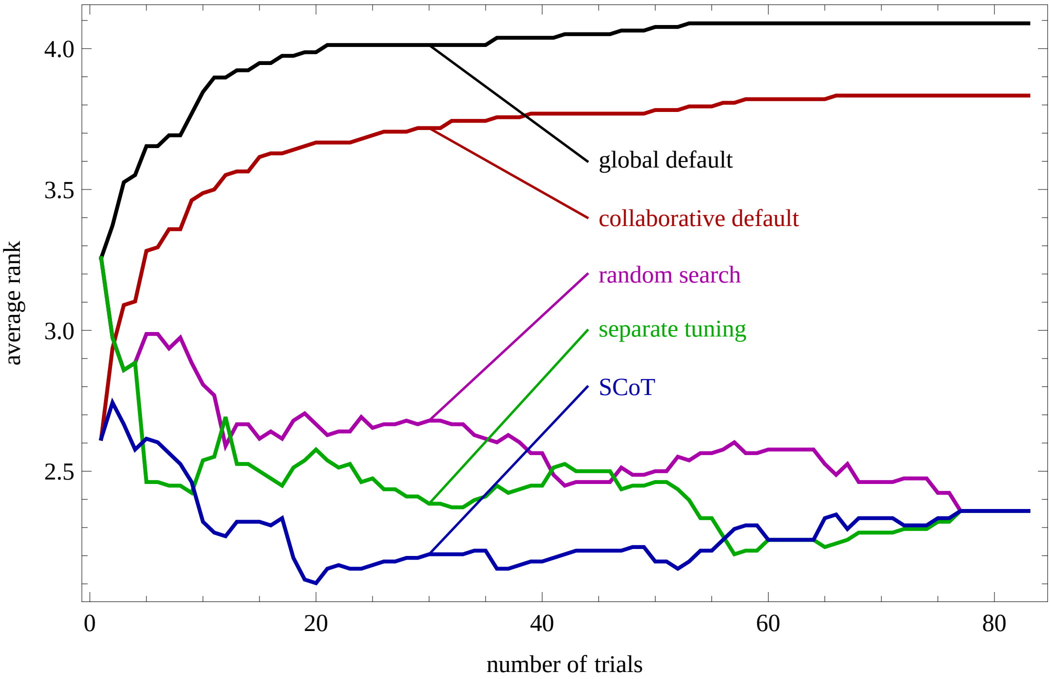


Figure 5.7: The average rank of the different methods as a function of the number of trials. Collaborative methods start from the value 2.62 and all non-collaborative methods start from 3.26, the first four iteration of separate tuning is the same as random search for reasons described in the text.

Part II

On adaptive MCMC algorithms, with applications to the Pierre Auger experiment

Chapter 6
The Pierre Auger experiment

Contents

The Pierre Auger observatory is a giant ultra-high energy cosmic ray detector located in the Argentinian pampa. Since we are members of the Pierre Auger collaboration, this experiment motivated a lot of our research, looping from modelling to the design of suitable inference algorithms to the application of these algorithms. This chapter does not contain personal contributions, but is rather devoted to presenting the Pierre Auger experiment, in order to motivate our contributions of Part II and introduce notions in view of Chapters 7. The target of this chapter is the non-physicist reader. The expert reader will find a more detailed and technical introduction in [92, Chapters 1 and 2], to which the present chapter owes a lot.

6.1 Ultra-high energy cosmic rays

The study of cosmic rays is a wide and active research topic, related to many fields of physics like particle and nuclear physics, astrophysics, and cosmology. Our present knowledge of elementary particles was initiated by the study of cosmic rays, with the discovery of the positron in 1932, the muon in 1937, the pion in 1947, etc. Despite the interest they raised and almost one century after their discovery, fundamental questions on cosmic rays remain unanswered, particularly at ultra-high energy: what kind of particles are these cosmic rays? If there exist sources, then where are they? How do the cosmic rays reach such high energies? What do they tell us about the cosmic accelerators producing such extreme energies? How strong are the magnetic fields that they go through on their way to Earth? How do they interact with the relic photons from the early universe that fill the universe, also known as the cosmic microwave background radiation (CMB)? What can we learn about particle interactions at these otherwise inaccessible energies? We will examine these different questions and recent progress towards answering some of them, including the last results from the Pierre Auger observatory, at the end of the chapter. In this section, after a brief historical introduction, we will have a closer look at cosmic showers, cascades of particles generated by cosmic rays hitting the atmosphere.

6.1.1 A brief history of cosmic rays

From balloon flights made by Victor Franz Hess as early as 1912 [73] it became clear that the ionization of the atmosphere increased very strongly with rising altitude. Hess noted that the intensity of the ionization first decreased, but that at around one kilometer it started to increase markedly. This would not be expected if the origin was entirely from radioactive decay of material in the Earth's crust, as had previously been suggested. Werner Kolhörster flew balloons to altitudes exceeding 9 kilometers in Germany and measured even higher ionization levels. Hess and Kolhörster concluded that the air was being ionized by an extraterrestrial source: cosmic rays. The term was coined by Robert Millikan who was trying to prove that cosmic rays were photons since they are very deeply penetrating in the atmosphere. The nature of this radiation remained unclear for many years, although it was shown early that the primary radiation included particles with energies as big as 10^{10} eV (electronvolts[1]), and also that the majority were positively charged.

In 1928, a particle detector with the ability to reveal the passage of charged particles was developed: the Geiger counter, developed by Hans Geiger, was used by Walther Bothe and Werner Kohlörster to prove that cosmic rays can penetrate thick absorbing materials. Bruno Rossi used three Geiger counters, disposing them on a horizontal surface, so that no single particle could pass through the three detectors. He developed a coincidence circuit to select only events triggering the three detectors at the same time. This apparatus measured a large number of coincidences proving in this way the existence of secondary particle showers: cascades of particles produced after the primary ray hit the atmosphere.

The phenomenon now known as extensive air showers (EAS) was discovered by Pierre Auger in 1938 in Paris [12]. Using two or three Geiger counters, operated in coincidence and separated by a variable distance of up to 300 meters, Auger

[1] The eV is an energy unit commonly used in particle physics, with $1\,\text{eV} \approx 1.6 \times 10^{-19}$ joules. For comparison, the energy released by nuclear fission of a heavy nucleus is a few 10^{10} eV, the most energetic proton collision the CERN Large Hadron Collider is planning to achieve is around 10^{17} eV, the Auger observatory is designed to gather data on cosmic rays of energy beyond 10^{18} eV, and the highest energy cosmic ray ever observed, the so-called *oh-my-god particle*, with its 3.2×10^{20} eV, is equivalent to the kinetic energy of a tennis ball served by professional player Maria Sharapova.

demonstrated that there are large particle showers arriving at ground level where particles are correlated in time and space. Pierre Auger estimated that the showers he detected were generated each by a single particle – the *primary particle* – hitting the atmosphere, with energies up to 10^{15} eV, a jump of five orders of magnitude over previous results.

The energy spectrum of cosmic rays – the empirical energy distribution of the primary particles – is depicted in Figure 6.1. One usually fits to this spectrum several power functions $E \mapsto E^{-\gamma}$, with changepoints known as the knee and the ankle, that can be seen in Figure 6.1. More precisely, the remarkable features of the spectrum are the following:

- the **knee** is thought to be an effect of the acceleration of cosmic rays by objects in our galaxy and marks the top energy that protons can reach when accelerated by a galactic supernova. The region around the knee is indicated in Figure 6.1. A less marked second knee around 10^{17} eV marks the top energy attainable by iron nuclei accelerated by a galactic supernova.
- the **ankle** at 3×10^{18} eV marks the transition energy above which the spectrum is dominated by cosmic rays of extragalactic origin, i.e. cosmic rays produced outside the milky way. The extragalactic mechanisms behind ultra-high energies are still widely discussed. Active galactic nuclei (AGNs), giant black holes at the center of certain galaxies like Centaurus A (the nearest extragalactic AGN to us), are credible candidates. The Pierre Auger observatory is designed to observe these ultra-high energies above the ankle, and the size of the observatory ($3\,000\,\mathrm{km}^2$) is explained by the rate of arrivals of such rays: one per square kilometer per year!
- the **cutoff** above 3×10^{19} eV, corresponding to the end of the cosmic ray spectrum. The cutoff can be explained through the interaction of cosmic rays with the cosmic background radiation, a process known as the Greisen-Zatsepin-Kuz'min (GZK) effect.

Data has been collected in the second half of the 20th century by always larger arrays of detectors, among which the Volcano Ranch array ($8\,\mathrm{km}^2$), the Haverah Park array ($12\,\mathrm{km}^2$), AGASA ($100\,\mathrm{km}^2$), and the Pierre Auger observatory ($3000\,\mathrm{km}^2$). Let us also mention HiRes, a fluorescence telescope. The Pierre Auger experiment targets the part of the spectrum above 10^{18} eV, the so-called *ultra-high energy cosmic rays*.

6.1.2 Looking inside an air shower

Cosmic rays are charged nuclei, believed to range from proton to iron in the classification table, with very high energies. When such a nucleus – the *primary particle* – hits the atmosphere, it interacts with the nuclei in air molecules, generating several *secondary particles*. Chances of hitting an nucleus in the air rise with energy, and

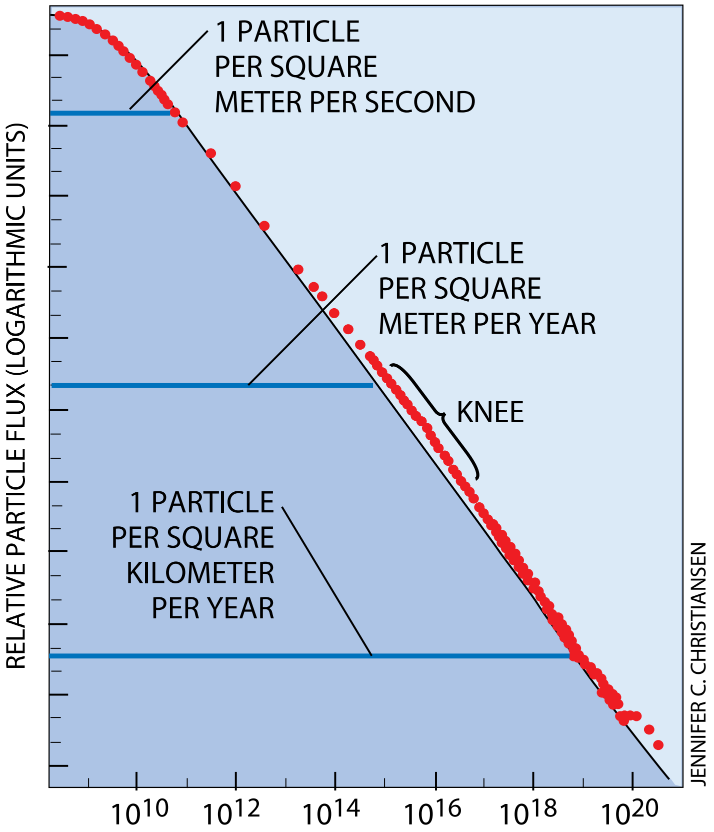


Figure 6.1: The cosmic ray energy spectrum. The black plain line corresponds to $E \mapsto E^{-3}$ and is here for reference.

also with the mass of the primary (protons are the lightest candidates for primaries, iron nuclei the heaviest). After a hit, the incoming energy is shared between the generated particles, which in turn interact in the atmosphere generating new particles. The repetition of this process produces a cascade of particles, the so-called *extensive air shower*.

Due to the steeply falling flux of incident particles with energy shown in Figure 6.1, direct observation of the primaries by high-altitude balloon and satellite experiments is only possible at low energies. At the high end of the spectrum, information regarding the energy, arrival direction and nature of the arriving particles can only be obtained through the observation of the cascade of secondary particles and its side effects.

A widely accepted model for extensive air showers is the Heitler model in Figure 6.2. The first interaction of the primary nucleus with air molecules gives rise to a shower in which we see three components: the nucleonic, pionic and electromagnetic cascade. The nucleonic cascade corresponds to the *leading* primary particle loosing its energy and other possible heavy product nuclei. The pionic cascade has its origin in the generated charged pions, unstable elementary particles that lose energy and progressively decay – spontaneously disintegrate – into muons, another elementary particle, often described as a very heavy analogue to the electron. There are also neutral pions created along the path, which are even more unstable and quickly disintegrate into electromagnetic particles (photons, electrons, positrons), leading to the electromagnetic cascade in Figure 6.2. The development of the shower is characterized by a growth phase, where interactions and generations of new particles dominate over decays, until the shower reaches its maximum at depth[2] $X_{\max}$. After that, because the falling particles have individually less and less energy and thus a higher probability to decay, decays dominate and the number of particles decreases. The depth $X_{\max}$ is highly correlated with the depth of the first interaction. It is thus a good indicator of the mass of the primary, since lighter particles such as protons statistically penetrate deeper in the atmosphere before hitting a nucleus than heavier particles like iron. Not shown on Figure 6.2 are various emissions of light that are not essential to our matter in this document, but one of them deserves to be mentioned for its importance in determining the energy of the primary: the *fluorescence light* comes from the interaction of low-energy electrons with the nitrogen molecules in the air. The excited molecules then radiate the acquired energy in the UV range and go back to their steady state.

EM particles interact more with the atmosphere than muons. Consequently, the longer the path of the shower in the atmosphere, the lower the proportion of EM particles in the shower at ground. At ground level, *inclined showers*, arriving almost tangentially to the surface of Earth, are essentially restricted to their muonic component.

6.1.3 Astrophysical questions raised by cosmic rays

The ultra-high energy cosmic rays (UHECRs) are the most energetic particles known. Understanding their origin, production and propagation can be expected to give us insight on basic aspects of our Universe.

6.1.3.1 Propagation of the cosmic rays

The propagation of cosmic rays is deeply related to the structure of our universe. On the one hand, cosmic rays are charged particles, so they are deflected by the magnetic fields they go through on their way, essentially in our galaxy. On the other hand, cosmic rays are energetic particles that interact with the other particles they encounter, like the photons of the CMB, and these interactions make

[2] Depths such as $X_{\max}$ are specified in $\mathrm{g\,cm^{-2}}$, since they are defined as the integral of the atmosphere density (in $\mathrm{g\,cm^{-3}}$) along the path taken (in cm). In other words, a depth means here the amount of atmosphere crossed. This unit allows to compare depths in different mediums.

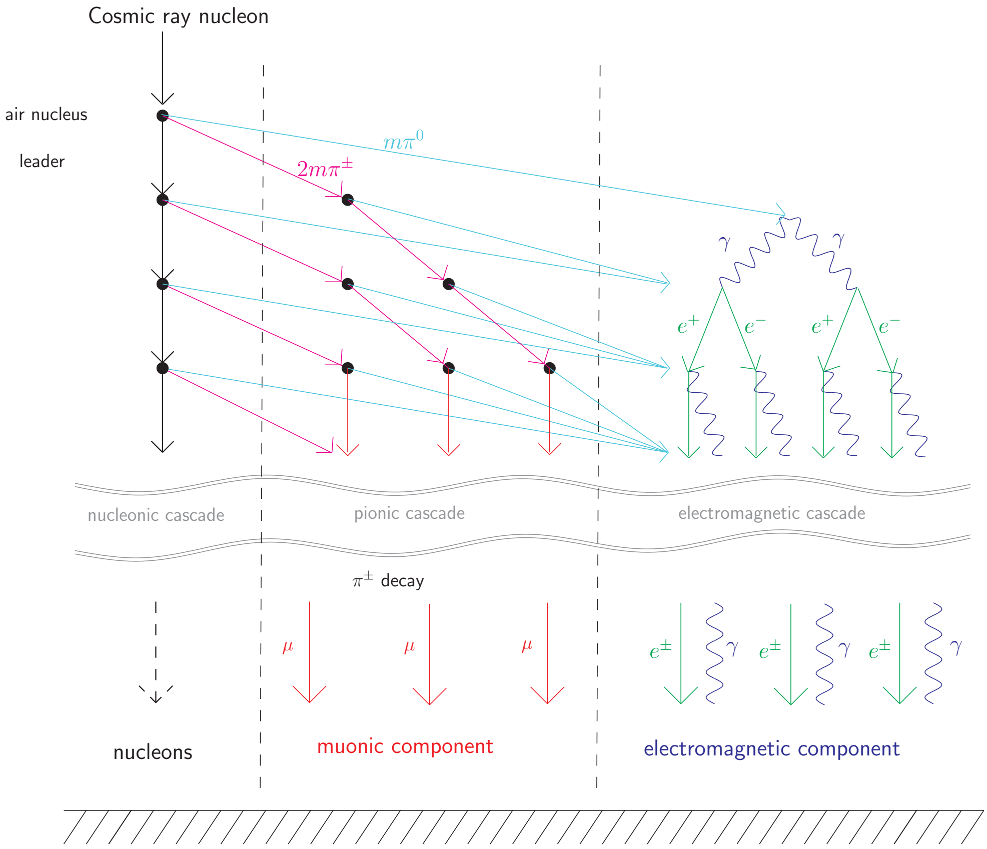


Figure 6.2: The Heitler model for an extensive air shower. Particles appearing are charged pions π^+ and π^-, neutral pions π^0, muons μ, electrons e^-, positrons e^+ and photons γ. See text for details.

each cosmic ray lose energy and possibly change its composition, while leaving its direction unchanged. Understanding the trajectories and energy distribution of cosmic rays thus constrains models of galactic and extragalactic fields and background radiations. For example, the experimental validation of the GZK effect (see Section 6.1.1) was a landmark in the understanding of the CMB.

6.1.3.2 Origin and acceleration of the ultra-high energy cosmic rays

It is particularly tough to determine the origin of UHECRs, but we can reasonably think that high energy protons, for instance, will not be deflected too much by the galactic magnetic fields, thus allowing us to pinpoint potential sources by estimating and drawing skymaps of the arrival directions of UHECRs.

Two classes of models were proposed for accelerating particles to such high energies: the so-called *top-down* and *bottom-up* models. Top-down models involve very heavy relics from the early universe decaying and require new physics, meaning that the current accepted model for elementary particles and their interaction – the *standard model* – is not sufficient to explain UHECRs under top-down production models. Bottom-up models are more popular, and explain UHECRs by extragalactic objects accelerating the particles along their path through different mechanisms that we will not delve in here. A good candidate for acceleration sites are the active galactic nuclei, supermassive black holes at the centers of certain galaxies.

Having a precise idea of the composition of the cosmic rays would also help in the search for sources: nuclei interact, for instance, differently with the CMB, and knowing in which proportion particles enter in the composition of cosmic rays would help setting limits on the distance at which we have to look for sources. Additionally, we know that iron nuclei deflect too much in our galaxy, even at the highest energies, so that a heavy composition would condemn source scans.

6.2 The Auger detector

We have seen the importance of estimating the energy spectrum, composition and arrival directions of UHECRs. To tackle these issues, the Pierre Auger experiment was conceived in the wake of always larger and more sophisticated detectors. It consists of two sets of devices: a *surface detector* (SD) and a *fluorescence detector* (FD). A summary drawing is shown in Figure 6.3.

The surface detector, shown in Figure 6.4, consists of 1 660 surface stations – water-filled tanks and their associated electronics – arranged on a triangular grid, the distance between two tanks being 1.5 kilometers, with the grid covering a total area of 3 000 square kilometers. On clear moonless nights, four optical stations look at the atmosphere above the array. Each station contains six telescopes, designed to detect the fluorescence light mentioned in Section 6.1.2.

6.2.1 The surface detector

When a charged particle crosses a medium – here water – at a greater speed than the speed of light in the medium (you can go faster than light in water!), then nature has the particle loose energy by emitting photons along its tracklength. This process is known as the *Cherenkov effect.* Recall the secondary particles reaching ground level are mainly muons, electrons, positrons and photons, see Figure 6.2). By virtue of the Cherenkov effect, muons, electrons and positrons generate photons in the water tanks of the SD. Individual photons are then seen by one of the three photodetectors (also called photomultiplying tubes or PMTs) in the tank. These photons are then converted to photoelectrons (electrons ejected from the detector atoms by the incoming photons). If the signal satisfies hardware triggering conditions designed by the experimenters to avoid recording noise, the current is then measured, integrated over bins of width 25 nanoseconds and discretized to produce

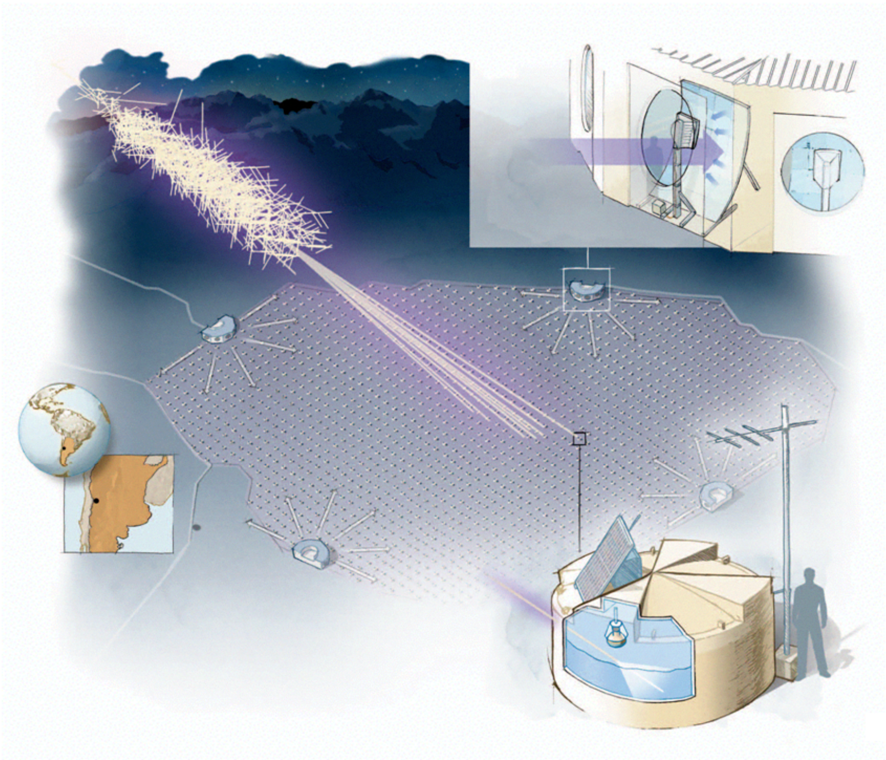


Figure 6.3: An overview of the Pierre Auger detector with its surface and fluorescence detectors. The main element of the SD is the tank (bottom right). The main element of the FD is the optical station (top right).

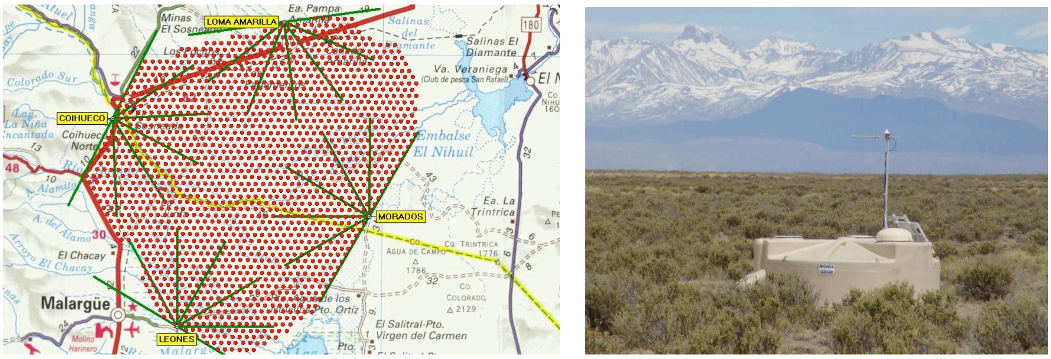


Figure 6.4: On the left panel, red dots illustrate the surface detector and its 1 660 water tanks spread over 3 000 square kilometers near the middle Andes, about 400 kilometers south of Mendoza, Argentina. Green lines indicate the field of view of each optical station of the fluorescence detector. A picture of an individual SD tank is shown on the right panel with the Andes in the background.

the final SD signal, the so-called FADC[3] histogram, or trace. The unit of the y-axis

[3] Flash analog to digital converter.

in FADC traces is the *vertical equivalent muon* (VEM, VEM-peak). It is a calibration unit that represents the mean signal of a muon crossing the tank vertically. Each PMT producing a histogram, there are three FADC histograms produced by each tank for each event. A view of a tank is given in Figure 6.4, while an example of FADC histogram is given in Figure 6.5. The analyses in this thesis deal with this SD signal.

6.2.2 The fluorescence detector

The four optical stations, the position of which is depicted in Figures 6.3 and 6.4 along with their fields of view, are similar to the one shown depicted in Figure 6.3. They are made of mirrors focusing the fluorescence light onto cameras. The FD is operated only during clear moonless nights (about 13% of the time) and thus accumulates less data than the SD, which works 24/7. FD data is essential in the estimation of the energy of the observed cosmic rays, and datasets consisting of events seen by both the SD and the FD are very valuable for the analysis.

6.2.3 Latest results

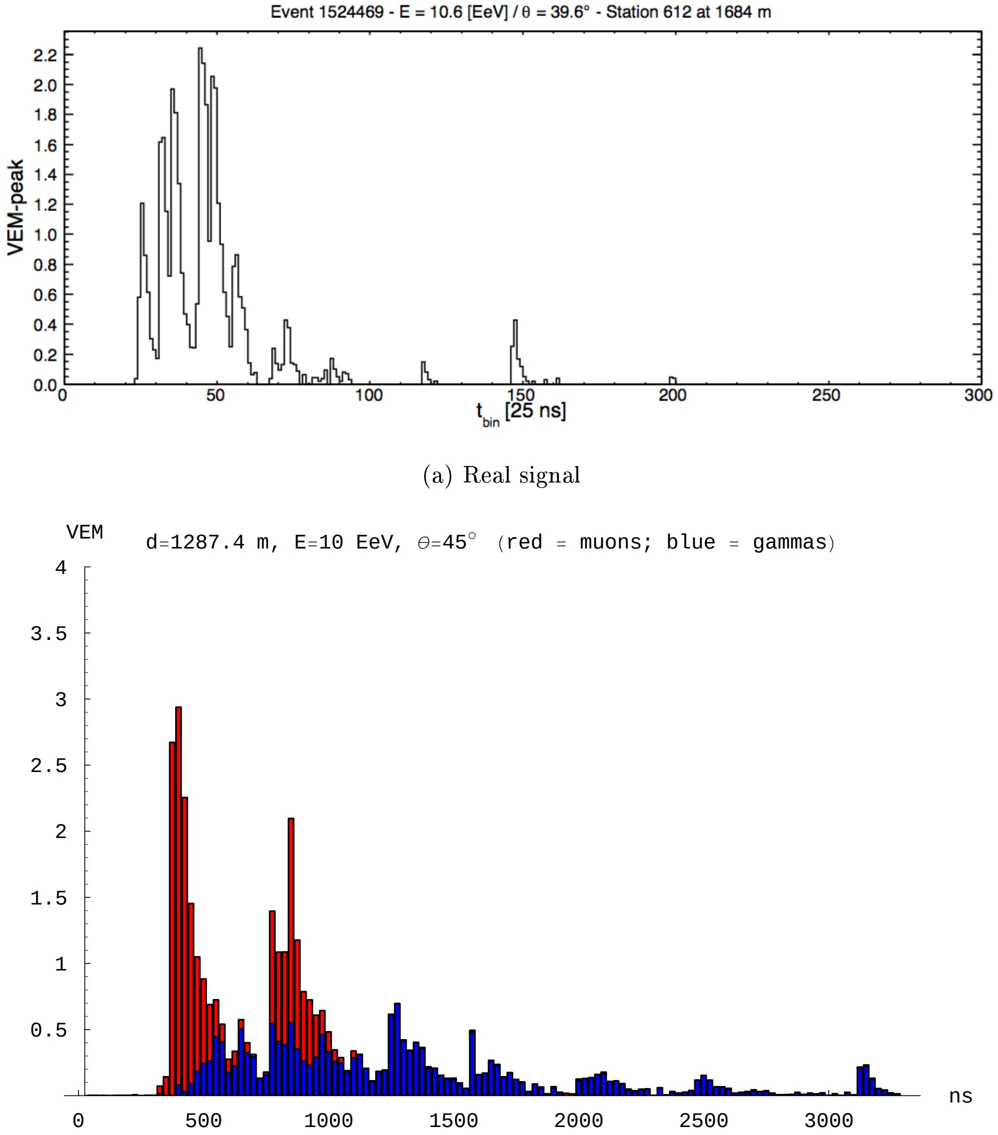
We quickly review here recent results obtained by the Auger collaboration on the energy spectrum, the mass composition and the potential sources of UHECRs.

6.2.3.1 Spectrum

Figure 6.6 presents the most recent spectrum published by the Auger collaboration [58]. The cutoff after $10^{19.4}$ eV is particularly pronounced. As previously mentioned, this cutoff is believed to be the result of the interaction of cosmic rays with the CMB (the GZK effect). However, other explanations are possible, and maybe complementary: it could be that cosmic rays cannot accelerate to higher energies for some reason. Furthermore, a GZK cutoff at $10^{19.4}$ eV is only compatible with a relatively light mass composition, while Auger seems to indicate a change to heavy composition in the highest energies, but how heavy?

6.2.3.2 Sources

Only the cosmic rays with the highest energies can be expected not to have been deflected by the galactic magnetic field. A skymap of the reconstructed arrivals of the most energetic events observed by Auger can be seen in Figure 6.7, which was published in [1]. Various statistical tests for correlation with astronomical catalogues are being run to detect potential sources. AGNs are plotted in blue on Figure 6.7, could they be the birth place of cosmic rays? The most recent test results lead to



(a) Real signal

(b) Simulated signal

Figure 6.5: FADC signal examples. Figure 6.5(a) shows a real FADC trace, while Figure 6.5(b) depicts a simulation where we separated the muonic and the EM component of the signal, according to what kind of secondary particle of the shower caused the photon emission. Muons (in red) typically produce peaky signals early in the trace, with an exponential tail. The abundant EM particles (in blue) generate a smoother and more elongated signal with lower amplitude.

rejecting the hypothesis of isotropy of the arrivals with an insufficient significance level to claim a discovery [1].

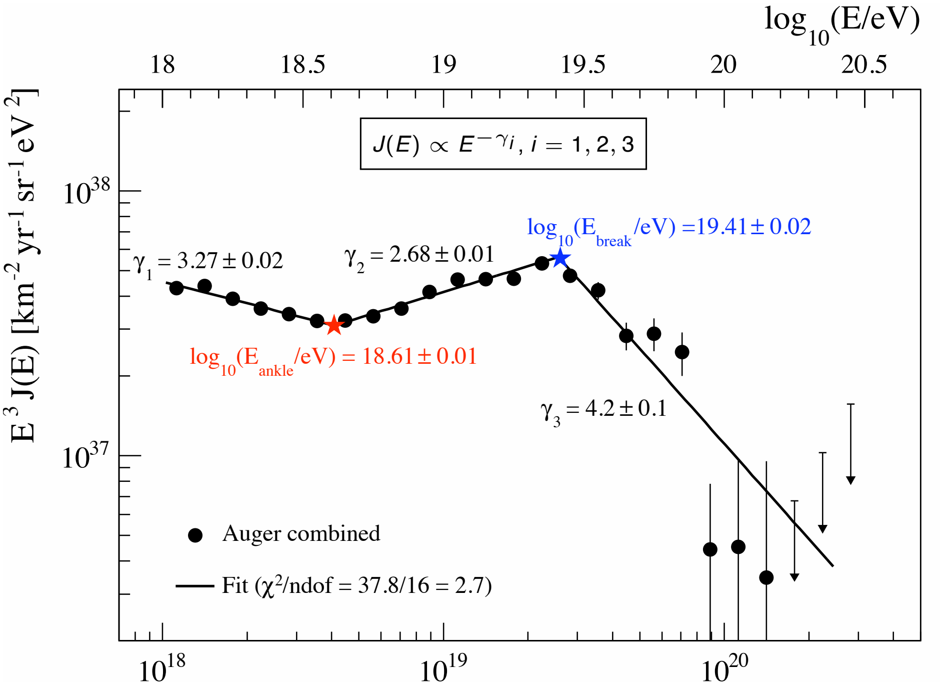


Figure 6.6: The latest spectrum published by the Auger collaboration [58]. Plotted is the flux multiplied by the cube of the energy, so that $E \mapsto E^{-3}$ would yield a flat horizontal line.

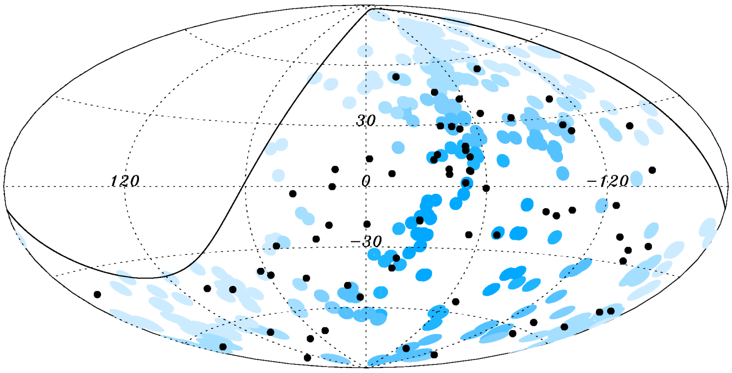


Figure 6.7: The 69 arrival directions of cosmic rays with energy larger than 5.5×10^{19} eV detected by the Pierre Auger observatory. They are plotted as black dots in an Aitoff-Hammer projection of the sky in galactic coordinates (our galaxy is on the x-axis). The solid line represents the border of the field of view of the Auger Observatory for zenith angles lower than 60° (0° corresponds to a vertical shower, 90° to a flat shower tangent to the surface of earth. Blue circles are centered at the position of the closest AGNs within the field of view of the observatory. Darker blue indicates larger relative exposure.

6.2.3.3 Mass composition

The Auger collaboration identified several independent observable quantities that are sensitive to the mass, and hence the nature, of the primaries. Some of these quantities are

- the average depth of the shower maximum $\langle X_{\text{max}} \rangle$ (see the Heitler model described in Section 6.1.2): the lighter the particle, the deeper it penetrates the atmosphere,
- the standard deviation $\text{RMS}(X_{\text{max}})$ is also correlated with the nature of the primary: the depth of proton showers tends to fluctuate more than the depth of iron showers,
- the number of muons that reached the ground and their average maximum muon production depth $\langle X^{\mu}_{\text{max}} \rangle$,
- the risetime asymmetry parameter Θ_{max}, an observable based on quantiles of the signal in different tanks.

These quantities are subject to independent systematic uncertainties (reconstruction uncertainty that will not decrease with more data). Due to the large fluctuations in shower development and the uncertainties in the interaction models, primary particle identification is very difficult and, as of today, not possible on an event-by-event basis. Figure 6.8 summarizes the latest results of the Auger collaboration on mass composition presented in [57]. Red lines indicate the expected results for pure proton showers, the lightest composition, while blue stands for pure iron, the heaviest. Different linestyles mean different *interaction models*, that is, different sets of rules and parameters to propagate individual particles in the shower. We will meet again two of these interaction models in Chapter 7: QGSJetII and EPOS. The description and the implementation of interaction models is an active research topic on its own, and astroparticle physicists usually compare to several of them, since all are plausible but different, through different choices of free parameters or different sampling algorithms for example. In particular, EPOS predicts more muons than QGSJetII, but both underestimate the number of muons in real data. All four plots in Figure 6.8, when compared to interaction models, indicate a change of composition towards heavy particles at high energies.

6.3 Conclusion and reading map

Cosmic rays convey important information on our universe. The Pierre Auger observatory gathers data on the cascade of particles each cosmic ray generates when entering the atmosphere, among which muons. Ad hoc procedures exist to estimate important quantities that are sensitive to key parameters like the energy, the composition and the arrival direction of the cosmic ray from the accumulated data. These estimates are often heavily dependent on existing simulators of the different interactions involved, the implementation of which is an active research topic on its own.

As often in experimental physics, convincing results will come from independent methods yielding consistent estimation of the key parameters. As statisticians, we think that a natural way to go back from data to the primary parameters is to derive a full generative model of a shower, with the numerous intermediate processes (particle interactions, propagation of the shower, detection) encoded as hidden – or *nuisance* – variables, and perform a full Bayesian analysis of the data with this model. Chapter 7 is devoted to the derivation of the "bottom part" of this generative model and a first approach to estimate the number of muons in a shower. The remaining chapters of Part II of this thesis deal with Markov chain Monte Carlo algorithms suitable for inference in this model.

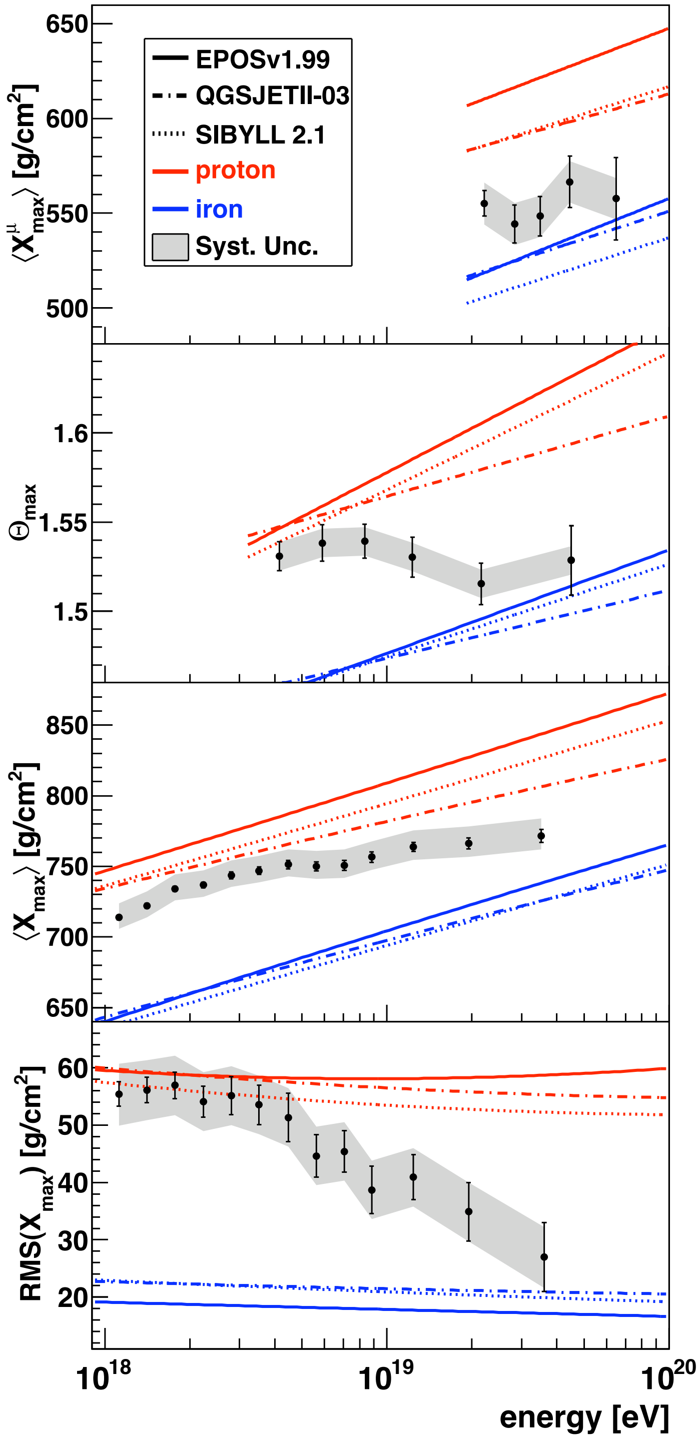


Figure 6.8: The latest results published by the Pierre Auger collaboration on mass composition [57]. The error bars correspond to the statistical uncertainty. Systematic uncertainty is represented by the shaded bands and is roughly to read as an additional uniform uncertainty which translates the precision of the observatory and reconstruction methods. Three different interaction models are plotted for reference.

Chapter 7
Inferring muons

Contents

In this chapter, we present our contributions to inference on the muonic content of cosmic showers. This is joint work with my advisor Balázs Kégl, and Darko Veberič (University of Nova Gorica, Slovenia) for the model in Chapter 7.2. These contributions were presented in [17] and [85].

7.1 Introduction

We reviewed in Chapter 6 the shower parameters that we are interested to infer with Auger data: composition, arrival direction, and energy of the primary cosmic ray. The most appealing way to reconstruct shower parameters is to write down a precise generative model of the Auger signal given these, and perform Bayesian inference. Designing a model and the associated computational tools is a long term task, of which Part II of this thesis shows different aspects. In this chapter, we start by deriving in Section 7.2 a model for the low-level SD tank signal of Auger presented in Section 6.2.1 that will motivate the methodological contribution of Chapter 9. Waiting for the overall model, the model of Section 7.2 could yield an efficient tankwise muon counter if used with an appropriate MCMC algorithm. To obtain physics results in the short term, we thus present in Section 7.3.2 an empirical Bayes procedure to estimate the number of muons in a shower given such a tankwise muon counter, and illustrate it on shower simulations. Finally, for a summary of the notations used here, we refer the reader to Appendix A.

7.2 A model for the Auger tank signal

The single muon response is a subject that was thoroughly explored in the early phase of the Auger collaboration [105, 54, 106, 48, 107, 26, 39, 119, 53, 55, 124, 56, 3, 52, 51, 5]. The main purpose of these studies was to understand the mean muon response in order to define an SD (surface detector, the tank array, see Chapter 6) energy estimate. Indeed energy is better estimated by the FD (fluorescence detector, see Chapter 6), but the latter is only operating on clear moonless nights, while the SD works 24/7. First, based on the muonic signal model, a calibration procedure was designed for estimating the total signal in individual tanks. Then the total signals were combined to compute one observable per shower, which was finally calibrated to the FD energy estimate.

The purpose of muon-counting [35, 61, 50, 62, 85] is to design a muon density estimator without the need for outside calibration. Our ultimate goal is similar to the program outlined in [56]: obtain a full parametrization of the muonic signal that can be used in a Monte Carlo Markov chain (MCMC; [110]) reconstruction approach as well as for fine-tuning the muon counting techniques. Beside this principal goal, we believe that this refined model may also help to improve the SD energy estimate (mainly by decreasing the statistical error on the individual shower estimates along the lines of [4]). The obtained model may also serve as a basis for a toy Monte Carlo tank simulator that can be used to quickly generate a large number of tank signals.

We concentrate in this work on vertical centered muons, that is, muons entering the tank vertically and through the center of its top face. Consequently, vertical muons have a tracklength of 1.2 meters, the height of a tank, and the three PMTs play a similar rôle. In Section 7.2, we formally describe the probability distribution of the FADC signal $\mathbf{x}$ of an individual muon given its time of arrival t_μ and the signal amplitude. The signal amplitude depends mostly on the tracklength L_μ of the muon in the tank, and, to a lesser extent, the energy of the muon E_μ. The energy dependence of the signal will be captured through a unitless *energy factor* ϕ_μ, which can be thought of as the ratio between the number of photons produced by the considered muon and the expected number of photons generated by a muon with kinetic energy of 1 GeV. Formally, the objective of this section is to develop a model for

$$p(\mathbf{x}|t_\mu, L_\mu, \phi_\mu) \ .$$

Let θ denote the zenith angle of a muon: θ is the angle of the arrival direction of the muon with respect to a vertical line. Vertical muons satisfy $\theta = 0$. The extension of the model to non-vertical muons will require the zenith-angle-dependent tracklength distribution that was already described in [87], and the distribution of the energy factor $p(\phi_\mu|E_\mu)$, which will be the subject of future work.

After summarizing our notations, we describe the model of the SD signal given the expected photoelectron count in Section 7.2.2. In Section 7.2.3, we describe the expected number of photoelectrons as a function of the muon and tank features, before concluding and relating this section to the rest of this thesis in Section 7.2.4.

7.2.1 The formal tank signal

Let $\mathbf{x} = (x_1, \ldots, x_N) \in \mathbb{R}^N$ be the FADC signal vector. We interpret x_i as the sum of the signals deposited by photoelectrons (PEs), corrupted by noise, in the time interval

$$[t_{i-1}, t_i) = [t_0 + (i-1)t_\Delta, t_0 + it_\Delta) \ , \tag{7.2.1}$$

where t_0 is the absolute starting time of the signal, and $t_\Delta = 25\,\text{ns}$ is the signal resolution (size of one bin). The signal is measured in FADC units. The goal is to parameterize the density

$$p(\mathbf{x}|t_\mu, L_\mu, \phi_\mu) \ , \tag{7.2.2}$$

where t_μ is the arrival time of the muon, L_μ is the tracklength of the muon, and ϕ_μ is a factor that captures the energy dependence of the signal amplitude. To fix the scale of ϕ_μ, we set $\phi_\mu \approx 1$ for an average[1] vertical atmospheric muon. In general, we will assume that $\max_{\phi_\mu} p(\phi_\mu) \approx 1$.

We will construct the model in a bottom-up fashion, that is, we start from the signal, and develop the model by adding explanatory parameters and nuisance parameters (Table 7.1).

7.2.2 The signal given the expected photoelectron count

First we will rewrite (7.2.2) as

$$\begin{aligned} p(\mathbf{x}|t_\mu, L_\mu, \phi_\mu) &= \int_{\mathbb{R}_+^N} p(\mathbf{x}, \bar{\mathbf{n}}|t_\mu, L_\mu, \phi_\mu)\,\mathrm{d}\bar{\mathbf{n}} \\ &= \int_{\mathbb{R}_+^N} p(\mathbf{x}|\bar{\mathbf{n}}, t_\mu, L_\mu, \phi_\mu) p(\bar{\mathbf{n}}|t_\mu, L_\mu, \phi_\mu)\,\mathrm{d}\bar{\mathbf{n}} \ , \end{aligned} \tag{7.2.3}$$

where $\bar{\mathbf{n}} = (\bar{n}_1, \ldots, \bar{n}_N) \in \mathbb{R}_+^N$, and $\bar{n}_i$ is the expected number of PEs in the bin $[t_{i-1}, t_i)$. Since the signal is independent of t_μ, L_μ, and ϕ_μ given $\mathbf{n}$, we can simplify (7.2.3) to

$$p(\mathbf{x}|t_\mu, L_\mu, \phi_\mu) = \int_{\mathbb{R}_+^N} p(\mathbf{x}|\bar{\mathbf{n}})\, p(\bar{\mathbf{n}}|t_\mu, L_\mu, \phi_\mu)\,\mathrm{d}\bar{\mathbf{n}} \ . \tag{7.2.4}$$

[1] More precisely, $\phi_\mu \approx 1$ for the *most likely* muon. Since $p(\phi_\mu)$ has a heavy tail and also a strong lower tail, in general $\mathbb{E}\{\phi_\mu\} \neq \max_{\phi_\mu} p(\phi_\mu)$.

Table 7.1: Explanatory (first 13 lines) and nuisance parameters. For explanatory parameters the fourth column is the prior distribution. For nuisance parameters we give the conditional distributions. A summary of notations can be found in Appendix A.

name	**not.**	**unit**	**law (priors or conditionals)**
bin width	t_Δ	ns	δ_{25}
signal decay time	τ	ns	δ_{60} or $\mathcal{N}_{60,5}$
signal risetime	t_d	ns	δ_4 or $\mathcal{N}_{4,1}$
signal start time	t_0	s	given for each signal
muon arrival time	t_μ	ns	inverse gamma $\mathcal{IG}_{\alpha,\beta}(t_\mu - t_0)$
muon tracklength	L_μ	m	$\delta_{1.2}$ (see [87] for inclined muons)
muon energy factor	ϕ_μ	unitless	δ_1 or $\mathcal{N}_{1,0.1}$ (see [16] for inclined muons)
muon amplitude	A_μ	unitless	$\phi_\mu L_\mu \nu$ (see Section 7.2.4)
avg number of PEs / 1 GeV / 1 m	ν	m^{-1}	δ_{228} or $\mathcal{N}_{228,120}$
PE signal mean (gain)	μ	adc	$\delta_{1.8}$ or $\Gamma_{18,0.1}$
PE signal shape	k	unitless	δ_2 or $\Gamma_{20,0.1}$
PE signal noise std	σ	adc	$\delta_{0.55}$ or $\mathcal{N}_{0.55,0.05}$
PE signal baseline	b	adc	$\mathcal{N}_{55,5}$
expected number of PEs in bin i	$\bar{n}_i$	unitless	$\phi_\mu L_\mu \nu \int_{t_{i-1}}^{t_i} p_{\tau,t_d}(t - t_\mu)\mathrm{d}t$ (see (7.2.21))
number of PEs in bin i	n_i	unitless	$\mathrm{Poi}_{\bar{n}_i}$
noiseless signal in bin i	$\bar{x}_i$	adc	$\Gamma_{n_i k,\mu/k}$
signal in bin i	x_i	adc	$\mathcal{N}_{b+\bar{x},\sigma}$

Since the second term is a Dirac delta – $\bar{\mathbf{n}}$ is a deterministic function of t_μ, L_μ, ϕ_μ, and some other explanatory parameters –, the convolution further simplifies to a simple product

$$p(\mathbf{x}|t_\mu, L_\mu, \phi_\mu) = p(\mathbf{x}|\bar{\mathbf{n}})\ \delta_{\bar{\mathbf{n}}(t_\mu,L_\mu,\phi_\mu)}\ . \tag{7.2.5}$$

The function $\bar{\mathbf{n}}(t_\mu, L_\mu, \phi_\mu)$ in the second term depends mainly on the time response profile which we will describe in Section 7.2.3. Here we start by developing the first term $p(\mathbf{x}|\bar{\mathbf{n}})$ of (7.2.5). First note that given the expected number of PEs, the bin-wise PE signals x_i are independent of each other. This is a quite realistic assumption since the signals vary due to some random events in the PMT, although some dependence can occur due to some secondary effects.[2] In this model we ignore these effects, and factor $p(\mathbf{x}|\bar{\mathbf{n}})$ to obtain

$$p(\mathbf{x}|\bar{\mathbf{n}}) = \prod_{i=1}^{N} p(x_i|\bar{n}_i)\ . \tag{7.2.6}$$

We now introduce a second nuisance parameter, the actual number of PEs n_i in the bin $[t_{i-1}, t_i)$, and rewrite $p(x_i|\bar{n}_i)$ as

$$p(x_i|\bar{n}_i) = \sum_{n_i=0}^{\infty} p(x_i, n_i|\bar{n}_i) = \sum_{n_i=0}^{\infty} p(x_i|n_i, \bar{n}_i)p(n_i|\bar{n}_i)\ . \tag{7.2.7}$$

[2] See Figure A-52 in [51].

Given n_i, the signal x_i is independent of the expected number of PEs $\bar{n}_i$, so $p(x_i|n_i, \bar{n}_i) = p(x_i|n_i)$. Since we can assume that the PEs were generated by capturing a small portion of Cherenkov photons, the number of PEs n_i is Poisson with parameter $\bar{n}_i$, and so (7.2.7) simplifies to

$$p(x_i|\bar{n}_i) = \sum_{n_i=0}^{\infty} p(x_i|n_i)\mathrm{Poi}_{\bar{n}_i}(n_i) \; . \tag{7.2.8}$$

The first term $p(x_i|n_i)$ depends on the spectrum of single PEs, the baseline, and the bin-wise noise. To model this convolution, we introduce our last nuisance parameter, the noiseless signal $\bar{x}_i$ in the bin $[t_{i-1}, t_i)$, and rewrite $p(x_i|n_i)$ as

$$p(x_i|n_i) = \int_0^{\infty} p(x_i, \bar{x}_i|n_i)\,\mathrm{d}\bar{x}_i = \int_0^{\infty} p(x_i|\bar{x}_i, n_i)p(\bar{x}_i|n_i)\,\mathrm{d}\bar{x}_i \; . \tag{7.2.9}$$

Given $\bar{x}_i$, the signal x_i is independent of the PE count n_i (it only depends on the noise and the baseline), so we can rewrite (7.2.9) as

$$p(x_i|n_i) = \int_0^{\infty} p(x_i|\bar{x}_i)p(\bar{x}_i|n_i)\,\mathrm{d}\bar{x}_i \; . \tag{7.2.10}$$

The bin-wise noiseless signal $\bar{x}_i$ is shifted by a baseline b, and also corrupted by an additive Gaussian noise with zero mean and standard deviation σ so the first term of (7.2.10) becomes

$$p(x_i|\bar{x}_i) = \mathcal{N}_{b+\bar{x}_i,\sigma}(x_i) \tag{7.2.11}$$

The noiseless signal $\bar{x}_i$ is a sum of n_i independent PE signals. It was measured [48, 63, 51] that single PEs deposit a signal which is almost exponential, having a relative variance

$$\frac{\text{variance}}{\text{mean}^2} \approx 0.5 \; . \tag{7.2.12}$$

The actual form of the distribution is not very important since the total signal is a sum of several PE signals, so we decided to model the single PE spectrum by a Gamma distribution with shape parameter $k = \text{mean}^2/\text{variance}$ and scale parameter $\theta = \text{variance}/\text{mean}$. We parametrize the distribution using the gain $\mu = k\theta$ and the shape parameter k. Since the distribution of the sum of n independent Gamma variates with parameters k and θ is $\Gamma_{nk,\theta}$, the distribution of the noiseless signal $\bar{x}_i$ given the number of PEs n_i is[3]

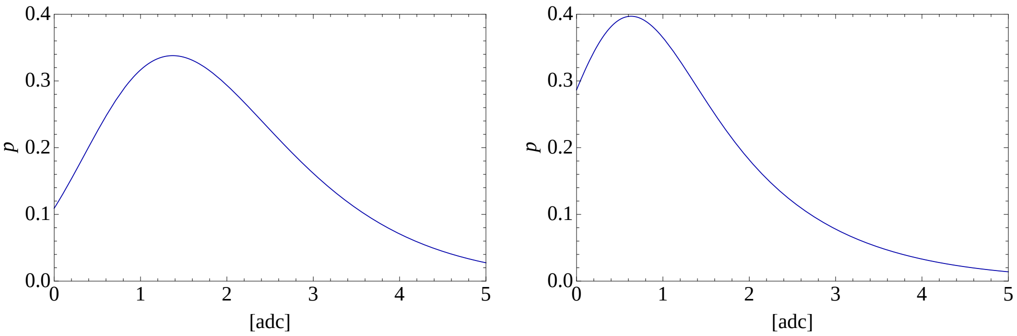
$$p(\bar{x}_i|n_i) = \Gamma_{n_i k,\mu/k}(\bar{x}_i) \; . \tag{7.2.13}$$

[3] To avoid the singularity, we will define $\Gamma_{0,\theta}(x) = \delta_0(x)$ where δ_a is the Dirac delta distribution centered at a.

Combining equations (7.2.8), (7.2.10), (7.2.11), and (7.2.13), our final bin-wise signal model is

$$\begin{aligned} p(x_i|\bar{n}_i) &= \sum_{n_i=0}^{\infty} \mathrm{Poi}_{\bar{n}_i}(n_i) \int_0^{\infty} \Gamma_{n_i k, \mu/k}(\bar{x}_i) \mathcal{N}_{b+\bar{x}_i,\sigma}(x_i)\, \mathrm{d}\bar{x}_i \\ &= \int_0^{\infty} \mathcal{N}_{b,\sigma}(x_i - \bar{x}_i) \underbrace{\sum_{n_i=0}^{\infty} \mathrm{Poi}_{\bar{n}_i}(n_i) \Gamma_{n_i k, \mu/k}(\bar{x}_i)}_{\text{compound Poisson}} \mathrm{d}\bar{x}_i \, . \qquad (7.2.14) \end{aligned}$$

Figure 7.1 shows the single PE spectrum under the model, using different values for the gain μ and the shape parameter k. The spectrum contains the additive noise, that is, we numerically integrated the convolution of (7.2.14) with $n_i = 1$. The curves are reasonably close to those depicted by Figure 2 in [63], Figures 1-2 in [48], or Figure A-53 in [51].



(a) $k = 2.5$, $\mu = 1.9$, $\sigma = 0.6$

(b) $k = 1.05$, $\mu = 1.2$, $\sigma = 0.6$

Figure 7.1: "Ideal" single PE spectra with additive noise.

The distribution $p(x_i|\bar{n}_i)$ is the convolution of a Gaussian and a compound Poisson – the latter being a sum of independent Gamma variables – so its mean is $b + \bar{n}_i\mu$ and its variance is

$$\bar{n}_i \mu^2 \left(1 + \frac{1}{k} + \frac{\sigma^2}{\bar{n}_i \mu^2}\right) . \qquad (7.2.15)$$

Setting aside the baseline translation, the relative variance is

$$\frac{1}{\bar{n}_i}\left(1 + \frac{1}{k} + \frac{\sigma^2}{\bar{n}_i \mu^2}\right) , \qquad (7.2.16)$$

where the third term in the parentheses can safely be ignored for large $\bar{n}_i$. The range of k goes from 1 (exponential PE spectrum) to ∞ (Dirac PE spectrum), with the relative signal variance (7.2.16) being between $1/\bar{n}_i$ (Dirac) and $2/\bar{n}_i$ (exponential).

Generating random variates from (7.2.14) is easy: one just has to go through the convolution chain. On the other hand, evaluating (7.2.14) given x_i and $\bar{n}_i$ (which we need in parameter estimation) can only be done numerically. One can approximate

(7.2.14) by simpler formulas, but we found that these approximations may corrupt the estimation of signal parameters μ, k, and σ. In case these parameters are known, we can approximate the convolution (7.2.14) by a Gamma distribution with the same mean and variance:

$$p(x_i|\bar{n}_i) \approx \Gamma_{k',\theta'}(x_i - b) \tag{7.2.17}$$

with

$$k' = \bar{n}_i \frac{k}{(k+1)\left(1 + \frac{k\sigma^2}{(k+1)\bar{n}_i\mu^2}\right)}\,, \tag{7.2.18}$$

$$\theta' = \mu\frac{k+1}{k} + \frac{\sigma^2}{\bar{n}_i\mu}\,. \tag{7.2.19}$$

If the signal parameters are estimated with a Monte Carlo Markov chain (MCMC) algorithm, we can explicitly introduce the nuisance parameters $\bar{n}_i$, n_i, and $\bar{x}_i$, and let the MCMC do the numerical integration.

7.2.2.1 Discretization

The FADC count is a discretized value of the original signal. The likelihood (7.2.14) is a good approximation of the real likelihood in case σ is larger than half the resolution (which is 1 in our case). In this case the discretization variance 1/12 is included in the estimated σ. In experiments we found that σ is around the critical value of 0.5, and we started to observe both a slight bias in the baseline estimate and a fluctuation of the noise estimate, depending where the real baseline was with respect to the FADC bin boundaries, so we opted to include the discretization in the likelihood by using

$$p(x_i|\bar{n}_i) = \sum_{n_i=0}^{\infty} \text{Poi}_{\bar{n}_i}(n_i) \int_0^{\infty} \Gamma_{n_i k,\mu/k}(\bar{x}_i) \int_{x_i-0.5}^{x_i+0.5} \mathcal{N}_{b+\bar{x}_i,\sigma}(x_i')\,\mathrm{d}x_i'\,\mathrm{d}\bar{x}_i\,. \tag{7.2.20}$$

instead of (7.2.14).

7.2.3 The distribution of the expected PE count in time

The second term $\bar{n}_i(L_\mu, \phi_\mu, t_\mu)$ of (7.2.5) determines the expected number of PEs in the bin $[t_{i-1}, t_i)$, given L_μ, ϕ_μ, and t_μ. We will use a simple model with three additional explanatory parameters, the risetime t_d, the rate of the exponential decay τ (both measured in ns), and the mean number ν of PEs generated by a muon with kinetic energy 1 GeV on a tracklength of 1 m. We found that this parametrization works fine for vertical centered muons, to which we limit the scope of this chapter. The refinement of the model for inclined non-centered muons, with treatment of the PMT asymmetries and direct light is postponed as future work.

Our model is based on the assumption that a muon generates a number of Cherenkov photons along its trajectory at a rate that depends on its energy. The photons are generated at a precise angle around the trajectory. They can be reflected several times on the walls of the tank before arriving into the PMT. It was measured that the photon distribution "mixes" (becomes uniform) after around two reflections within another 5 to 10 ns. This means that the risetime is in the first two bins (at most). After that the rate of arrival in the PMT becomes exponential because of the constant decay rate due to the absorption of photons in the water and reflection losses. The rate can change from one tank to another, and it was observed to change also in time. To model the risetime, we assume that the photon generation is uniform in a window of width t_d. The decay phase is modeled by an exponential with parameter τ. The convolution can be solved analytically to obtain the time response distribution

$$p_{\tau,t_d}(t) = \frac{1}{t_d} \cdot \begin{cases} 0 & \text{if } t < 0 \ , \\ 1 - \exp(-t/\tau) & \text{if } 0 \leq t < t_d \ , \\ \exp\big(- (t - t_d)/\tau \big) - \exp(-t/\tau) & \text{if } t_d \leq t \ . \end{cases} \tag{7.2.21}$$

Figure 7.2(a) shows the time response distribution with typical parameter values $\tau = 60\,\text{ns}$ and $t_d = 4\,\text{ns}$.

To obtain the expected number of PEs in a bin, we first have to integrate $p_{\tau,t_d}(t)$ in the bin, and then we have to multiply it with (a) the tracklength L_μ, (b) the average number of PEs per unit tracklength $\nu = 228\,\text{m}^{-1}$, and (c) the energy factor ϕ_μ,

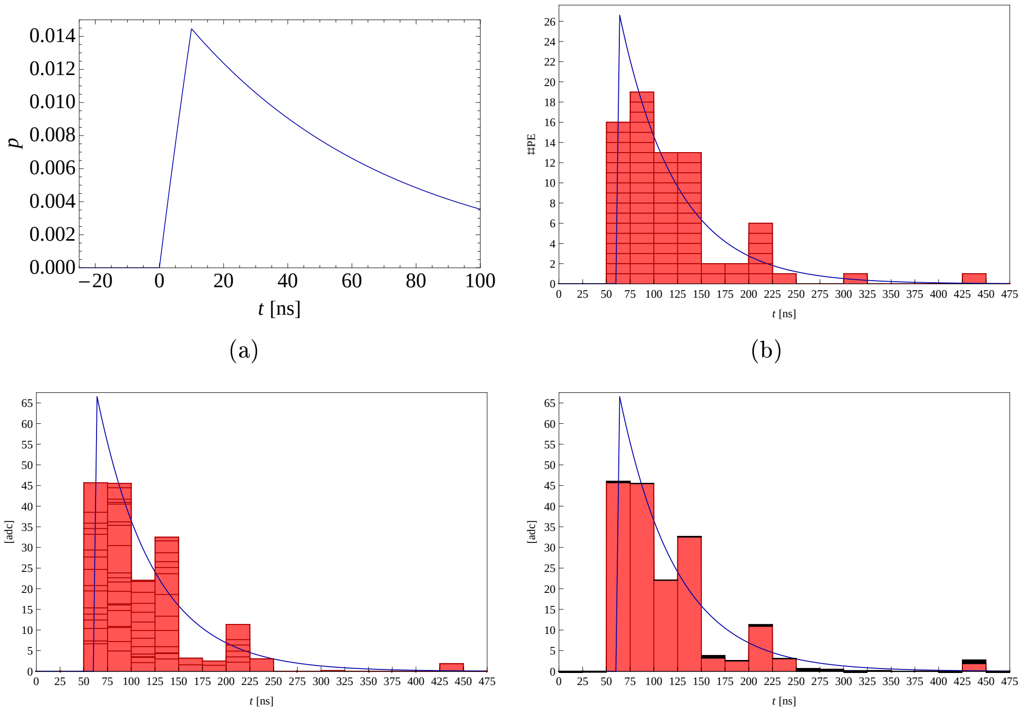
$$\bar{n}_i(L_\mu, \phi_\mu, t_\mu) = \phi_\mu L_\mu \nu \int_{t_{i-1}}^{t_i} p_{\tau,t_d}(t - t_\mu)\,\text{d}t \ . \tag{7.2.22}$$

7.2.4 Priors and features of the tank signal model

Note that the tracklength L_μ, the energy factor ϕ_μ, and the average number of PEs per unit tracklength ν appear only in a product in , which is not a problem here since L_μ is equal to the height of a tank and ν is constant, but will make them hard to disentangle for muons which are not vertical. We thus group these parameters and define the amplitude $A_\mu = \phi_\mu L_\mu \nu$ of a muon. The prior on A_μ is thus $\mathcal{N}_{\nu L_\mu,1}$ for vertical muons, according to Table 7.1. An example prior for A_μ in the case of inclined muons can be found in [16].

The prior on t_μ is taken to be inverse gamma, according to personal communications with Sylvie Dagoret-Campagne (Université Paris-Sud XI), with parameters that depend on several higher-level geometric features of the shower. For simulations in this thesis, we use an $\mathcal{IG}_{2,100}$ prior.

The model is summarized in the Directed Acyclic Graph (DAG) given in Figure 7.3, along with Table 7.1. Although we have considered N_μ fixed throughout the section,



(a)

(b)

(c)

(d)

Figure 7.2: (a) The time response distribution with typical values $\tau = 60\,$ns and $t_d = 4\,$ns. (b) The same ideal time response, with an example $\mathbf{n}$ vector. (c) Example gamma vector $\bar{x}$ vector. (d) Example x vector with Gaussian noise depicted in black.

we specify in the DAG a Poisson prior $\mathrm{Poi}_{\overline{N}_\mu}$ for N_μ to allow estimation of N_μ via reversible jump MCMC algorithms [68].

Given N_μ and, the muon arrival times $\mathbf{t} = (t_1, \ldots, t_{N_\mu})$ and amplitudes $\mathbf{A} = (A_1, \ldots, A_{N_\mu})$, the likelihood of a signal $\mathbf{x}$ is finally

$$p(\mathbf{x}|\mathbf{A}, \mathbf{t}, N_\mu) = \prod_{i=1}^{N} p(x_i|\bar{n}_i(\mathbf{A}, \mathbf{t}, N_\mu)) \;, \tag{7.2.23}$$

where $p(x_i|\bar{n}_i)$ is defined in (7.2.14) and

$$\bar{n}_i(\mathbf{A}, \mathbf{t}, N_\mu) = \sum_{j=1}^{N_\mu} A_j \int_{t_{i-1}}^{t_i} p_{\tau,t_d}(t - t_j)\, \mathrm{d}t \;.$$

Priors are specified in Table 7.1, so that we can speak of the posterior distribution of $\mathbf{A}, \mathbf{t}$. The posterior is likely to show correlation between variable blocks A_i, t_i belonging to different muons, since, for instance, two muons close in time will "compete" to explain the signal and thus have negatively correlated amplitudes. When it comes to choosing the right MCMC algorithm to sample from this posterior, this

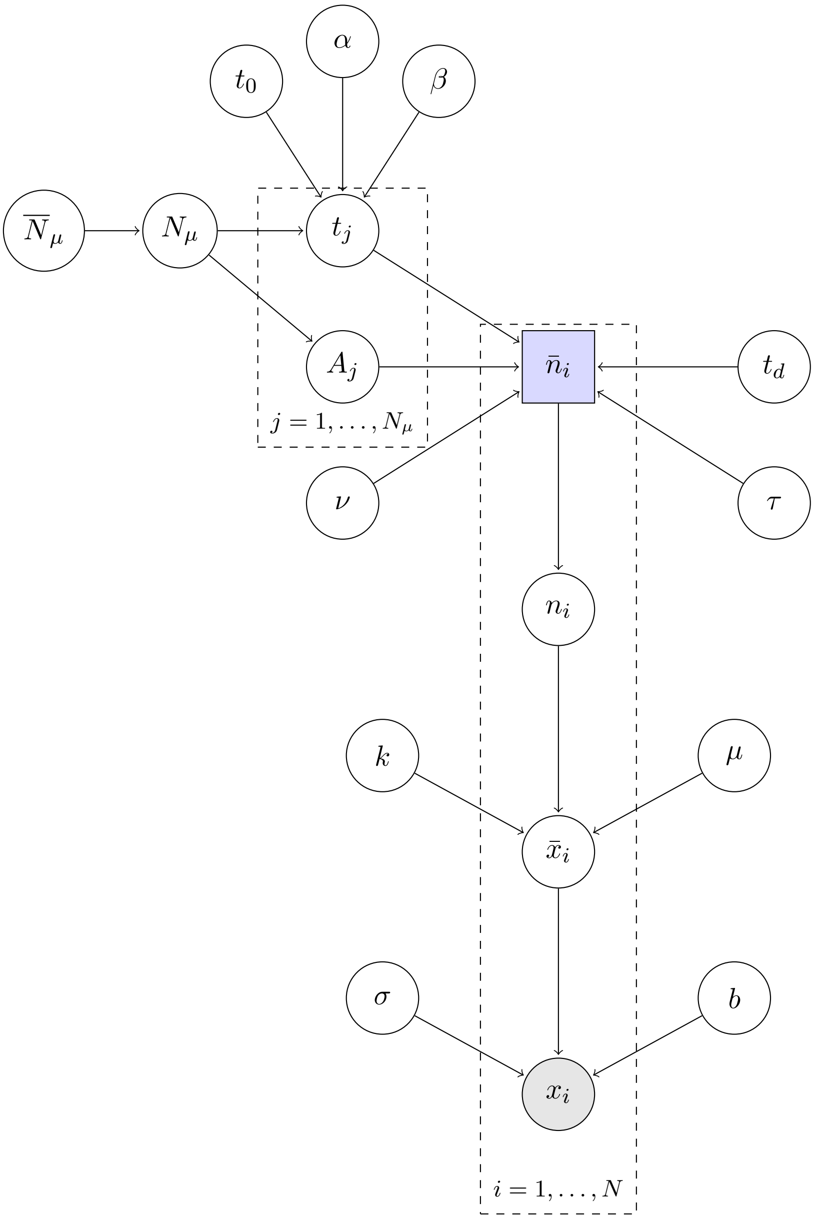


Figure 7.3: Directed acyclic graph summarizing the tank signal model described in Section 7.2. Distributions are specified in Table 7.1. Data $\mathbf{x}$ is in gray, while $\mathbf{n}$, depicted in a blue rectangle, is a deterministic function of its parent nodes.

motivates the use of adaptive MCMC algorithms that learn the covariance of the target distribution on the fly and use it to propose points more efficiently.

Another feature of the model (7.2.23) is permutation-invariance. Indeed, in the case where $N_\mu = 2$, for instance, it comes

$$p(\mathbf{x}|(A_1, A_2), (t_1, t_2), N_\mu = 2) = p(\mathbf{x}|(A_2, A_1), (t_2, t_1), N_\mu = 2) \; .$$

If the chosen prior does not favor any permutation of the muons either, then MCMC inference in such a model has to be done with care since label switching will occur, as explained in Chapter 8, where we also review existing algorithmic solutions. Chapters 9 and 10 are devoted to the presentation, application and theoretical analysis of AMOR, an adaptive algorithm that solves the label switching problem and was motivated by the model presented in this section.

7.3 Going large-scale: counting muons in a shower

Deriving a full generative model for cosmic showers is a difficult task: after the low-level tank signal of Section 7.2, it remains to add the EM component to the model in a cheap-to-sample-from way, derive the parts of the model that govern the shower generation 1) between the tank and the production site of the particles and 2) between production and the hit of the primary ray. Furthermore, once derived a full model, a classical treatment would require to have an MCMC algorithm that integrates over all nuisance variables. For *one single tank*, the model presented in Section 7.2 already involves several nuisance variable vectors of length the number of FADC bins, which will be a few hundreds for real data. In a joint work with A. Roodaki (Télécom ParisTech), we have preliminary experiments that show that a clever design of RJMCMC proposals yields good results on the single tank model, but nothing guarantees that once we go large-scale and put all tanks together in a model that includes more high level nuisance variables, we will be able to implement an efficient sampler. This is why we now present a tractable way of performing inference on the number of muons in a shower, an important observable as seen in Chapter 6, that requires only a *tankwise* posterior on the number of muons given the signal. Once it will contain the EM component, an RJMCMC algorithm applied to the model of Section 7.2 will fill that rôle. Note that methods presented in this Section 7 are not novel, our contribution is their application to Auger simulations and data[4].

In Section 7.3.1, we describe the Lateral Distribution Function (LDF) and precise what we mean by estimating the number of muons. In Section 7.3.1.3, we present the shower likelihood and priors, before deriving an empirical Bayes algorithm to estimate the LDF parameters in Section 7.3.2. In Section 7.3.2.1, we benchmark our method with a Bayesian neural network as tankwise muon counter. We showed in [85] that this approach outperforms the current state-of-the-art implemented in Auger software.

[4] Since the use of Auger data would imply this thesis is not in open access, we will only consider official simulations here. We refer interested readers with Auger collaboration rights to [85].

7.3.1 The lateral distribution function

The "number of muons" in a shower is an ill-defined notion: is it the number of muons produced, the number of muons reaching the ground, the number of muons observed? The Auger collaboration defined observables to quantify the muonic content of a shower.

Let the *shower axis* be an imaginary line that goes through the first interaction point of the primary particle and follows the arrival direction. Let the shower *core* be the intersection of the shower axis and the ground. Denote by r the distance of a tank to the shower axis. The *Lateral Distribution Function* (LDF) is the glue that links tankwise results. It is defined as the number of muons in an imaginary tank at distance r from the shower axis. The LDF is typically given a simple parametric form; once its parameters are estimated – we say the LDF is *reconstructed* – using tankwise estimates of the number of muons, we can compute, for instance, the number $N_\mu(1000)$ of muons in an imaginary tank at $r = 1000\,\mathrm{m}$ as a function of the energy of the shower and its inclination.

7.3.1.1 Notations for data

Assume we have n showers, indexed by i. If the shower is simulated, then its *real* energy E_i and zenith angle θ_i are provided. The ith shower has m_i active detectors, indexed by j. Out of these m_i detectors, k_i have triggered and $m_i - k_i$ are non-triggering: data is censored. Without loss of generality, we will assume that the first k_i have triggered and the last $m_i - k_i$ are non-triggering. The real number of muons in the jth detector of the ith shower will be denoted by $N_{i,j}$ (we omit the index $_\mu$ for simplicity). $N_{i,j}$ is, of course, only observed in simulations, and we use it only for sanity checks of the approach presented in this section, such as checking the LDF estimation is working well when provided with the real numbers of muons. For each detector, we also observe its distance from the shower axis $r_{i,j}$ and an indicator $\iota_{i,j}$ whether it has triggered or not (that is, $\iota_{i,j} = 1$ if the detector triggers, 0 if it is non-triggering). When we talk about the whole shower i, the vector of the numbers of muons in all detectors will be denoted by $\mathbf{N}_i$. Similarly, the vector of distances to the shower axis, the vector of FADC signals and the vector of trigger indicators will be $\mathbf{r}_i$, $\mathbf{X}_i$, and $\boldsymbol{\iota}_i$, respectively.

In the final application, E_i and θ_i will be estimated by existing ad-hoc techniques (the FD estimate of E is, for example, based on the quantity of fluorescence light emitted), and $N_{i,j}$ will be replaced by the probability table $p_{\mathrm{TE}}(N|\mathbf{x}_{i,j})$ provided by a tankwise muon counting algorithm that is assumed to be given. Such a tankwise counter can be an RJMCMC algorithm applied to the model of Section 7.2, or a regressed estimator as in [85] and Section 7.3.2.1.

7.3.1.2 The LDF parametrization

We use a log-log parabola for the LDF parametrization. The three parameters of the fit are the number $N_\mu(1000)$ of muons at 1000 m, the slope of the fit β and the curvature parameter γ. To simplify the notation, we will use ν for $\log N_\mu(1000)$, so the LDF function is

$$\overline{N}_{\mathrm{LDF}}(r,\nu,\beta,\gamma) = \exp\left(\nu + \beta \log\frac{r}{1000\,\mathrm{m}} + \gamma \log^2\left(\frac{r}{1000\,\mathrm{m}}\right)\right). \tag{7.3.1}$$

We use the notation $\overline{N}_{\mathrm{LDF}}$ to emphasize that the function expresses the expected number of muons in a detector at a distance r from the shower axis. Accordingly, the parameter $\exp(\nu) = N_\mu(1000)$ will target the expected number of muons in an imaginary detector at 1000 m from the shower axis.

7.3.1.3 The shower likelihood

To simplify the notation, in some of the formulas we will omit the dependence of $\overline{N}_{\mathrm{LDF}}$ on r, ν, β, and γ. Given the lateral distribution function $\overline{N}_{\mathrm{LDF}}(r,\nu,\beta,\gamma)$, the likelihood of the number of muons is a simple Poisson

$$p(N|\overline{N}_{\mathrm{LDF}}) = \mathrm{Poi}_{\overline{N}_{\mathrm{LDF}}}(N)\ .$$

We assume that the probability that a detector triggers

$$p(\iota = 1|N) = f_\iota(N)$$

is a deterministic non-parametric function of the number of muons in the detector. This is definitely a simplification and it could be refined in a subsequent analysis. We do not parametrize f_ι, rather, we provide it as a table of probabilities, see Section 7.3.2. The trigger probability in a detector given the fit is then

$$\begin{aligned} p(\iota = 1|r,\nu,\beta,\gamma) &= p(\iota = 1|\overline{N}_{\mathrm{LDF}}(r,\nu,\beta,\gamma)) \\ &= \sum_N p(\iota = 1|N)p(N|\overline{N}_{\mathrm{LDF}}) \\ &= \sum_N f_\iota(N)\mathrm{Poi}_{\overline{N}_{\mathrm{LDF}}}(N)\ , \end{aligned}$$

where we use the simplifying assumption in the second equality. Similarly, the no-trigger probability is

$$\begin{aligned} p(\iota = 0|r,\nu,\beta,\gamma) &= p(\iota = 0|\overline{N}_{\mathrm{LDF}}(r,\nu,\beta,\gamma)) \\ &= \sum_N p(\iota = 0|N)p(N|\overline{N}_{\mathrm{LDF}}) \\ &= \sum_N (1 - f_\iota(N))\mathrm{Poi}_{\overline{N}_{\mathrm{LDF}}}(N)\ . \end{aligned} \tag{7.3.2}$$

Since we do not observe the number of muons in non-triggering detectors, (7.3.2) is itself the likelihood of a detector being non-triggering given the fit and the distance r of the detector from the shower axis. Given that we observe the *real* number of muons N in the detector, the likelihood of the pair (ι, N) of a detector that has triggered is

$$\begin{aligned} p_{\text{real}}(\iota = 1, N | r, \nu, \beta, \gamma) &= p(\iota = 1, N | \overline{N}_{\text{LDF}}(r, \nu, \beta, \gamma)) \\ &= p(\iota = 1 | N, \overline{N}_{\text{LDF}}) p(N | \overline{N}_{\text{LDF}}) \\ &= p(\iota = 1 | N) p(N | \overline{N}_{\text{LDF}}) \\ &= f_\iota(N) \text{Poi}_{\overline{N}_{\text{LDF}}}(N) \ . \end{aligned}$$

In this case the likelihood of the triplet ν_i, β_i, γ_i is

$$\begin{aligned} p_{\text{real}}(\mathbf{N}_i, \boldsymbol{\iota}_i, \mathbf{r}_i | \nu_i, \beta_i, \gamma_i) = &\prod_{j=1}^{k_i} f_\iota(N_{i,j}) \text{Poi}_{\overline{N}_{\text{LDF}(r_{i,j}, \nu_i, \beta_i, \gamma_i)}}(N_{i,j}) \\ &\times \prod_{j=k_i+1}^{m_i} \sum_N (1 - f_\iota(N)) \text{Poi}_{\overline{N}_{\text{LDF}(r_{i,j}, \nu_i, \beta_i, \gamma_i)}}(N) \ . \end{aligned} \tag{7.3.3}$$

When using the probability table $p_{\text{TE}}(N|\mathbf{x})$ provided by the tankwise estimator, the likelihood of the pair of the trigger indicator ι and the FADC signal $\mathbf{x}$ is also a sum

$$p_{\text{TE}}(\iota = 1, \mathbf{x} | r, \nu, \beta, \gamma) = \sum_N p_{\text{TE}}(N|\mathbf{x}) f_\iota(N) \text{Poi}_{\overline{N}_{\text{LDF}}}(N) \ ,$$

so the likelihood of the triplet ν_i, β_i, γ_i is

$$\begin{aligned} p_{\text{TE}}(\mathbf{X}_i, \boldsymbol{\iota}_i, \mathbf{r}_i | \nu_i, \beta_i, \gamma_i) = &\prod_{j=1}^{k_i} \sum_N p_{\text{TE}}(N|\mathbf{x}_{i,j}) f_\iota(N) \text{Poi}_{\overline{N}_{\text{LDF}(r_{i,j}, \nu_i, \beta_i, \gamma_i)}}(N) \\ &\times \prod_{j=k_i+1}^{m_i} \sum_N (1 - f_\iota(N)) \text{Poi}_{\overline{N}_{\text{LDF}(r_{i,j}, \nu_i, \beta_i, \gamma_i)}}(N) \ . \end{aligned} \tag{7.3.4}$$

7.3.1.4 The priors (constraints)

Maximizing the likelihoods (7.3.3) or (7.3.4) with completely free parameters is possible, but it can lead to degenerate fits with large uncertainties especially for lower energy events or events with bad geometry, that is, showers with “unlucky” patterns of triggered tanks. As an alternative, one could fix β and/or γ and fit only the log number of muons ν at 1000 m, but this would eliminate the shower-to-shower fluctuation of β and γ and bias the fits unnecessarily for high quality events. In this analysis we opt for a best-of-both-worlds solution: we define “soft constraints” that can be formally interpreted as priors over the parameters ν, β, and γ.

In particular, the parameters ν, β, and γ will be modeled with independent Gaussians

$$\begin{aligned} p(\beta|\mu_\beta, \sigma_\beta) &= \mathcal{N}_{\mu_\beta, \sigma_\beta} \ , \\ p(\gamma|\mu_\gamma, \sigma_\gamma) &= \mathcal{N}_{\mu_\gamma, \sigma_\gamma} \ , \\ p(\nu|\mu_\nu, \sigma_\nu) &= \mathcal{N}_{\mu_\nu, \sigma_\nu} \ . \end{aligned} \tag{7.3.5}$$

We know by experience that all parameters depend on the energy and the zenith angle of the shower. This dependence is very strong for ν, but β and γ can also have slight but clear trends in energy and zenith angle. We thus parametrize their means and standard deviations in E and θ. In this setup, the likelihood of the ith shower (conditioned on μ_β, μ_γ, μ_ν, σ_β, σ_γ, and σ_ν) is

$$\begin{aligned} & p_{\text{real}}(\mathbf{N}_i, \boldsymbol{\iota}_i, \mathbf{r}_i, \nu_i, \beta_i, \gamma_i | \mu_\beta, \mu_\gamma, \mu_\nu, \sigma_\beta, \sigma_\gamma, \sigma_\nu) = \\ & \qquad p_{\text{real}}(\mathbf{N}_i, \boldsymbol{\iota}_i, \mathbf{r}_i | \nu_i, \beta_i, \gamma_i) \times \mathcal{N}_{\mu_\beta(E_i,\theta_i), \sigma_\beta(E_i,\theta_i)}(\beta_i) \\ & \qquad \times \mathcal{N}_{\mu_\gamma(E_i,\theta_i), \sigma_\gamma(E_i,\theta_i)}(\gamma_i) \times \mathcal{N}_{\mu_\nu(E_i,\theta_i), \sigma_\nu(E_i,\theta_i)}(\nu_i) \ , \end{aligned} \tag{7.3.6}$$

when the number of muons is observed in each detector, and

$$\begin{aligned} & p_{\text{TE}}(\mathbf{X}_i, \boldsymbol{\iota}_i, \mathbf{r}_i, \nu_i, \beta_i, \gamma_i | \mu_\beta, \mu_\gamma, \mu_\nu, \sigma_\beta, \sigma_\gamma, \sigma_\nu) = \\ & \qquad p_{\text{TE}}(\mathbf{X}_i, \boldsymbol{\iota}_i, \mathbf{r}_i | \nu_i, \beta_i, \gamma_i) \times \mathcal{N}_{\mu_\beta(E_i,\theta_i), \sigma_\beta(E_i,\theta_i)}(\beta_i) \\ & \qquad \times \mathcal{N}_{\mu_\gamma(E_i,\theta_i), \sigma_\gamma(E_i,\theta_i)}(\gamma_i) \times \mathcal{N}_{\mu_\nu(E_i,\theta_i), \sigma_\nu(E_i,\theta_i)}(\nu_i) \ , \end{aligned} \tag{7.3.7}$$

when the number of muons is estimated using the tankwise estimator.

In our first attempt, the mean functions $\mu_\beta(E, \theta)$, $\mu_\gamma(E, \theta)$, and $\mu_\nu(E, \theta)$ were simple polynomial parametrizations in E and θ, but we found that the natural shape of some of these functions did not follow any simple polynomial, so this rigid setup either led to unnecessary biases or the degrees of the polynomials had to be unreasonably high. To overcome this problem, we settled in a nonparametric solution in which overall trends were modeled with low-order polynomials, and the residuals were then fitted with smooth nonparametric function using Gaussian processes (see [108, 86] and Section 2.2.2). The standard deviation functions $\sigma_\beta(E, \theta)$, $\sigma_\gamma(E, \theta)$, and $\sigma_\nu(E, \theta)$ are linear functions of E and θ.

7.3.2 An empirical Bayes setup

We now detail how the empirical Bayes framework [33] applies to our reconstruction. In a classical Bayesian analysis, the goal would be to draw inference on $\boldsymbol{\phi}$ by means of the posterior distribution $p(\boldsymbol{\phi}|\mathbf{data})$. In our case $\mathbf{data}$ represents either $\big\{(\mathbf{N}_i, \boldsymbol{\iota}_i, \mathbf{r}_i)\big\}_{i=1}^n$ or $\big\{(\mathbf{X}_i, \boldsymbol{\iota}_i, \mathbf{r}_i)\big\}_{i=1}^n$, and the parameter vector $\boldsymbol{\phi}$ is $\big\{(\nu_i, \beta_i, \gamma_i)\big\}_{i=1}^n$. Since events (showers) are independent, the likelihood $p(\mathbf{data}|\boldsymbol{\phi})$ is simply the product of (7.3.3) or (7.3.4) for all events. In our case, the prior $p(\boldsymbol{\phi})$ is further parametrized by

$$\boldsymbol{\xi} = (\mu_\beta, \mu_\gamma, \mu_\nu, \sigma_\beta, \sigma_\gamma, \sigma_\nu) \ .$$

In the empirical Bayes setup, we do not fix the prior, rather we estimate it by maximizing the *marginal likelihood*

$$p(\mathbf{data}|\boldsymbol{\xi}) = \int p(\mathbf{data}, \boldsymbol{\phi}|\boldsymbol{\xi})\mathrm{d}\boldsymbol{\phi} = \int p(\mathbf{data}|\boldsymbol{\phi})p(\boldsymbol{\phi}|\boldsymbol{\xi})\mathrm{d}\boldsymbol{\phi} \ , \tag{7.3.8}$$

where we used the fact that, given $\boldsymbol{\phi}$, the **data** is independent of the hyperparameters $\boldsymbol{\xi}$. Once the *maximum marginal likelihood estimator* (MMLE)

$$\widehat{\boldsymbol{\xi}} = \arg\max_{\boldsymbol{\xi}} p(\mathbf{data}|\boldsymbol{\xi})$$

is found, the showerwise parameters $\boldsymbol{\phi} = \{\phi_i\}_{i=1}^n = \big\{(\nu_i, \beta_i, \gamma_i)\big\}_{i=1}^n$ can be estimated by using the distributions $p(\phi_i|\text{data}_i, \widehat{\boldsymbol{\xi}})$.

Maximizing (or even computing) (7.3.8) is intractable even if we suppose that the underlying densities factorize and the factors $p(\text{data}_i|\phi_i)$ and $p(\phi_i|\boldsymbol{\xi})$ have simple forms (Gaussians, for instance). To overcome this problem, we use a well-known trick in statistics that builds on the concavity of the log function and Jensen's inequality to define the lower bound

$$\log \int p(\mathbf{data}, \boldsymbol{\phi}|\boldsymbol{\xi})\, \mathrm{d}\boldsymbol{\phi} \geq \int q(\boldsymbol{\phi}) \log \frac{p(\mathbf{data}, \boldsymbol{\phi}|\boldsymbol{\xi})}{q(\boldsymbol{\phi})}\, \mathrm{d}\boldsymbol{\phi} = \mathcal{F}(q, \boldsymbol{\xi}) \ . \tag{7.3.9}$$

Note that the inequality is true for *any* density q over the shower parameters $\boldsymbol{\phi} = \big\{(\nu_i, \beta_i, \gamma_i)\big\}_{i=1}^n$ with an appropriate support. We alternately optimize $\mathcal{F}(q, \boldsymbol{\xi})$ in its two parameters, in what could be called an expectation-maximization algorithm (EM; [49]) with Laplace approximation. We now quickly describe these two steps.

1. In the *E-step*, we fix the hyperparameters $\boldsymbol{\xi}^{(t)}$, and optimize $\mathcal{F}(q, \boldsymbol{\xi})$ in q. Computing $p(\mathbf{data}, \boldsymbol{\phi}|\boldsymbol{\xi}) \propto p(\boldsymbol{\phi}|\mathbf{data}, \boldsymbol{\xi}^{(t)})$ is still infeasible, so we first approximate it. Since the shower events are independent, the posterior factorizes so that we can fit each shower independently:

   $$p(\boldsymbol{\phi}|\mathbf{data}, \boldsymbol{\xi}^{(t)}) = \prod_{i=1}^n p(\phi_i|\text{data}_i, \boldsymbol{\xi}^{(t)}) \propto \prod_{i=1}^n p(\text{data}_i|\phi_i)p(\phi_i|\boldsymbol{\xi}^{(t)}) \ , \tag{7.3.10}$$

   where the general term of the last product is exactly our likelihood (7.3.6) or (7.3.7) with fixed constraints $\boldsymbol{\xi}^{(t)} = \{\mu_\bullet^{(t)}, \sigma_\bullet^{(t)}\}_{\bullet=\beta,\gamma,\nu}$. We now approximate $p(\phi_i|\text{data}_i, \boldsymbol{\xi}^{(t)})$ with a Gaussian centered at its maximum with covariance matrix an estimate of the inverse Hessian[5] at this maximum:

   $$p(\phi_i|\text{data}_i, \boldsymbol{\xi}^{(t)}) \approx \mathcal{N}_{\widehat{\phi}_i, \widehat{\Sigma}_i}(\phi_i) \ , \tag{7.3.11}$$

   Now putting this approximation back in (7.3.10) and (7.3.9), and upon noting that $\mathcal{F}(q, \boldsymbol{\xi})$ is minus the Kullback-Leibler divergence between q and the

[5] We will see that because of the uncorrelated prior on ξ, it is actually enough to consider diagonal Σ_i's, but we stay general here at no cost.

approximate $p(\phi_i|\mathrm{data}_i, \boldsymbol{\xi}^{(t)})$, $\mathcal{F}(q, \boldsymbol{\xi})$ is maximized by setting q to the same approximation:

$$q^{(t)}(\boldsymbol{\phi}) = \prod_{i=1}^{N} \mathcal{N}_{\widehat{\phi}_i, \widehat{\Sigma}_i}(\phi_i) \; . \tag{7.3.12}$$

2. In the *M-step*, we fix the posterior $q^{(t)}(\boldsymbol{\phi})$ and maximize $\mathcal{F}(q^{(t)}, \boldsymbol{\xi})$ (7.3.9) in its second input to obtain

$$\begin{aligned} \boldsymbol{\xi}^{(t+1)} &= \arg\max_{\boldsymbol{\xi}} \mathcal{F}(q^{(t)}, \boldsymbol{\xi}) \\ &= \arg\max_{\boldsymbol{\xi}} \int q^{(t)}(\boldsymbol{\phi}) \log p(\mathbf{data}, \boldsymbol{\phi}|\boldsymbol{\xi}) \, \mathrm{d}\boldsymbol{\phi} \; . \end{aligned}$$

Since $q^{(t)}(\boldsymbol{\phi})p(\mathbf{data}|\boldsymbol{\phi})$ does not depend on $\boldsymbol{\xi}$, this further simplifies to

$$\boldsymbol{\xi}^{(t+1)} = \arg\max_{\boldsymbol{\xi}} \int q^{(t)}(\boldsymbol{\phi}) \log p(\boldsymbol{\phi}|\boldsymbol{\xi}) \, \mathrm{d}\boldsymbol{\phi} \; . \tag{7.3.13}$$

By (7.3.12) and (7.3.5), this is equivalent to the three independent following optimizations:

$$\begin{aligned} \boldsymbol{\xi}^{(t+1)} &= \arg\min_{\mu_\beta, \sigma_\beta} \sum_{i=1}^{n} \log \sigma_\beta(E_i, \theta_i) + \frac{1}{2} \frac{(\mu_\beta(E_i, \theta_i) - \widehat{\beta}_i^{(t)})^2 + \widehat{\sigma}_{\beta_i}^{(t)^2}}{\sigma_\beta(E_i, \theta_i)^2} \; , \\ (\mu_\gamma^{(t+1)}, \sigma_\gamma^{(t+1)}) &= \arg\min_{\mu_\gamma, \sigma_\gamma} \sum_{i=1}^{n} \log \sigma_\gamma(E_i, \theta_i) + \frac{1}{2} \frac{(\mu_\gamma(E_i, \theta_i) - \widehat{\gamma}_i^{(t)})^2 + \widehat{\sigma}_{\gamma_i}^{(t)^2}}{\sigma_\gamma(E_i, \theta_i)^2} \; , \\ (\mu_\nu^{(t+1)}, \sigma_\nu^{(t+1)}) &= \arg\min_{\mu_\nu, \sigma_\nu} \sum_{i=1}^{n} \log \sigma_\nu(E_i, \theta_i) + \frac{1}{2} \frac{(\mu_\nu(E_i, \theta_i) - \widehat{\nu}_i^{(t)})^2 + \widehat{\sigma}_{\nu_i}^{(t)^2}}{\sigma_\nu(E_i, \theta_i)^2} \; , \end{aligned} \tag{7.3.14}$$

where $\widehat{\phi_i}^{(t)} = (\widehat{\beta}_i^{(t)}, \widehat{\gamma}_i^{(t)}, \widehat{\nu}_i^{(t)})^T$ and $\mathrm{Diag}(\widehat{\Sigma_i}^{(t)}) = (\widehat{\sigma}_{\beta_i}^{(t)^2}, \widehat{\sigma}_{\gamma_i}^{(t)^2}, \widehat{\sigma}_{\nu_i}^{(t)^2})$. The minimizations can be done over any suitable family of functions. In [85], we used kernel regression with polynomial trends for the means $\mu_\bullet(E, \theta)$ and linear functions for the standard deviations $\sigma_\bullet(E, \theta)$.

Finally, the trigger probability table $f_\iota(N)$ can, in principle, be also optimized in the E-step. However, in our applications, we found that letting $f_\iota(N)$ free gave too much flexibility to the fit, so we decided to fix it to $\{f_\iota(0) = 0, f_\iota(1) = 0.1, f_\iota(2) = 0.6, f_\iota(3) = 0.9, f_\iota(4) = 1.0, \ldots\}$. These values were set manually, trying to minimize the bias of the final estimator of $N_\mu(1000)$. Among the three shower parameters, only the curvature γ happened to be sensitive to the actual values in the probability table which was one of the reasons we kept γ free.

7.3.2.1 Illustrative examples

We applied the described empirical Bayes estimation of the LDF parameters in [85]. The tankwise posterior p_{TE} on the number of muons was a Bayesian neural network [25] trained on simulations (approx. 18 000 showers, 150 000 individual tank signals).

Figure 7.4 shows the effect of the EM iterations on some example reconstructions. The first row contains "nice" events with detectors spaced uniformly around 1000 m and with good detectorwise estimations of the number of muons. These events are reconstructed well in the first iteration, so the gradually tightening prior or constraint (7.3.5) has no significant effect on the estimated $\widehat{N}_\mu(1000)$.

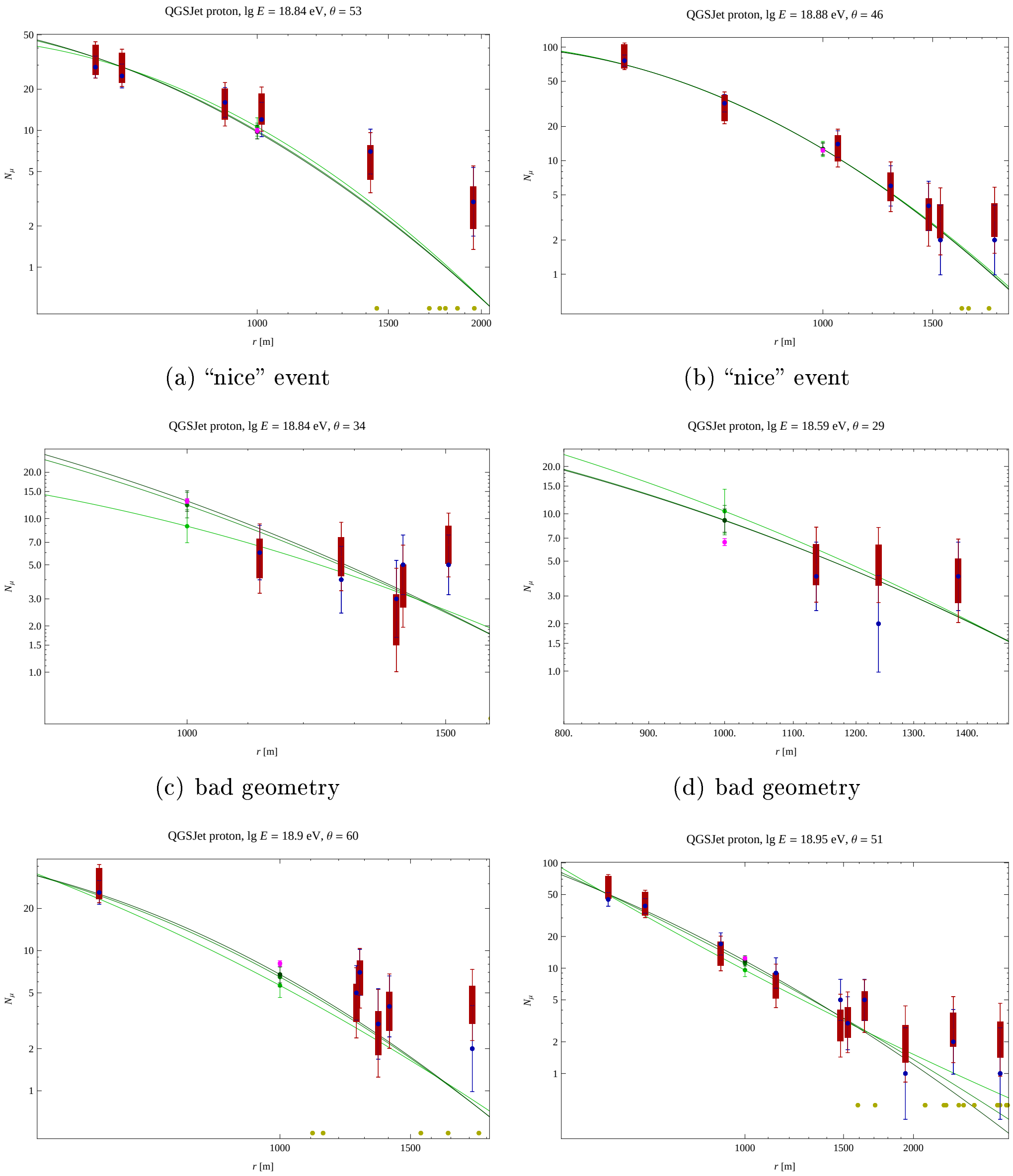
The second row contains events with bad geometry: the detector near the shower core is saturated, so all usable detectors are further from shower core than 1000 m. In the standard LDF reconstruction, these events profit from the fixed slope. The EM rounds have a similar effect here: the slope and the curvature is constrained by the tightening prior (7.3.5). The advantages are that 1) we do not have to fix the slope beforehand, the average slope (given the energy and the zenith angle is estimated from data), and 2) we do not fix the slope at all, allowing for shower-to-shower fluctuation. "Nice" but fluctuating events profit from this additional flexibility.

The third row contains some "exotic" events from the point of view of reconstruction. Inclined events (Figure 7.4(e)) often have non-triggering detectors downstream relatively close to the shower core. The upstream and downstream detectors usually average out, but, depending the actual geometry, the shape of unconstrained fit might be significantly altered, biasing also the $\widehat{N}_\mu(1000)$ estimate. The solution to this problem would be a parametrization that can account for the asymmetry. In the meantime, pulling the slope and curvature back towards the mean using our iterative reconstruction can correct this bias.

The detectorwise estimator is far from being perfect, but the estimation error usually cancels out when the event contains a relatively large number of detectors. However, in some unlucky cases, the estimation error can have a "structure". Figure 7.4(f) depicts an event where the number of muons are underestimated near 1000 m and overestimated at the edges of the fit, causing the unconstrained (log-log) LDF to be convex. Again, the effect of the data-dependent priors (7.3.5) is that the curvature is pulled back towards the population mean, greatly improving the $\widehat{N}_\mu(1000)$ estimate.

7.4 Conclusion

Building on previous studies, we have derived in Section 7.2 the first complete generative model for the muonic tank traces. To be applicable to data, this model should now incorporate the electromagnetic component of the tank signal, which is smoother and of smaller amplitude, as shown in Figure 6.5(b). EM particles generate a signal similar to the muons, but their number is typically bigger (hundreds against a few tens at most for muons) and their amplitude and time of arrival distributions are significantly different. Since there are so many EM particles and since we do not need to estimate precisely the features of each of them, we are looking for a cheaper-to-estimate EM model, with less nuisance variables. The choice of this model, along with an efficient RJMCMC algorithm reconstructing the sum of the muonic and EM components, are current joint work with Alireza Roodaki (Télécom ParisTech). Once we will have the posterior on the number of muons in each tank, we will in a first time estimate the muonic content of each shower with the empirical Bayes algorithm of Section 7.3.1. Benchmarked with a discriminative algorithm providing the tankwise posteriors, this empirical Bayes procedure was demonstrated to be promising. In the near future, we plan to continue to develop the model for cosmic showers, going from the tank to the production of the particles near the shower axis, and then from the production to the primary hit on the atmosphere.



(a) “nice” event

(b) “nice” event

(c) bad geometry

(d) bad geometry

(e) inclined, non-triggering detectors, asymmetry

(f) correcting the detectorwise

Figure 7.4: Example reconstructions on simulations. The blue dots are real number of muons N_μ with $\sqrt{N_\mu}$ Poisson error bars. The red bars are one-sigma regions of the estimated number of muons $\widehat{N}_\mu = \mathbb{E}\left\{p_{\mathrm{TE}}(N|\mathbf{x})\right\}$ with $\widehat{\sigma}_{N_\mu} = \sqrt{\mathrm{Var}\left\{p_{\mathrm{TE}}(N|\mathbf{x})\right\}}$ of the artificial neural network (ANN), and the red error bars are the complete approximate error bars $\sqrt{\widehat{N}_\mu + \widehat{\sigma}^2_{N_\mu}}$. The yellow dots are non-triggering detectors. The magenta dot is the real number of muons at 1000 m, available since we are using simulations. The green curves are the LDF fits: light green depicts the first empirical Bayes iteration, green the second, dark green the third. The green dots and error bars are the corresponding $\widehat{N}_\mu(1000)$ estimates.

Chapter 8

A review on relabeling MCMC algorithms

Contents

Motivated by the study of the Auger tank model in Section 7.2, we review in this chapter the label switching problem and existing relabeling MCMC algorithms. We will benchmark some of these algorithms on an illustrative example in Chapter 9. Other reviews can be found in [123, 80].

8.1 The label switching problem

Markov chain Monte Carlo (MCMC) is a generic approach for exploring complex probability distributions based on sampling. It has become the *de facto* standard tool in many applications of Bayesian inference. However, a very common situation in which MCMC algorithms face serious difficulties is when the target distribution is known to be invariant under some permutations (or block permutations) of the variables. In that case, the difficulties are both computational, as most often the MCMC algorithm fails to validly visit all the modes of the posterior, but also inferential, in particular rendering marginal posterior inference about the individual variables particularly cumbersome [37]. This latter difficulty is usually referred to as the *label switching problem* in the literature [123]. The most well-known example of this situation arises when performing Bayesian inference in a mixture model. In this case the mixture likelihood is invariant to permuting the mixture components, and the prior itself often does not favor any specific ordering of the mixture component [36, 123, 79, 80, 101, 121, 94]. Another example of importance arises in signal processing with additive decomposition models. In this case, the observed signal is represented as the superposition of individual signals, and the main goal is to recover the individual signals or their parameters and most often to determine the number of individual signals that are present [116, 115, 16]. The Auger tank signal model presented in Section 7.2 is such an additive model.

We now illustrate the label switching problem on the core of the Auger tank signal model of Section 7.2. Consider four centered vertical muons with different arrival times t_i and multiplicative amplitudes $A_1, \dots, A_4$, generating the Poisson signal $\mathbf{n}$ described in Section 7.2. We do not further include here the subsequent steps of the model, since at this stage, the likelihood

$$p(\mathbf{n}|\mathbf{t}, \mathbf{A})$$

is already invariant to permutations of the four muons. Figure 8 depicts the results of applying an adaptive MCMC [69] to estimate the parameters A_i, t_i, $i = 1, \dots, 4$. Label-switching can be seen in Figure 8.1(a) through the constant change of rôle played by each marginal chain, as well as in Figure 8.1(b) through the multimodality of marginal histograms.

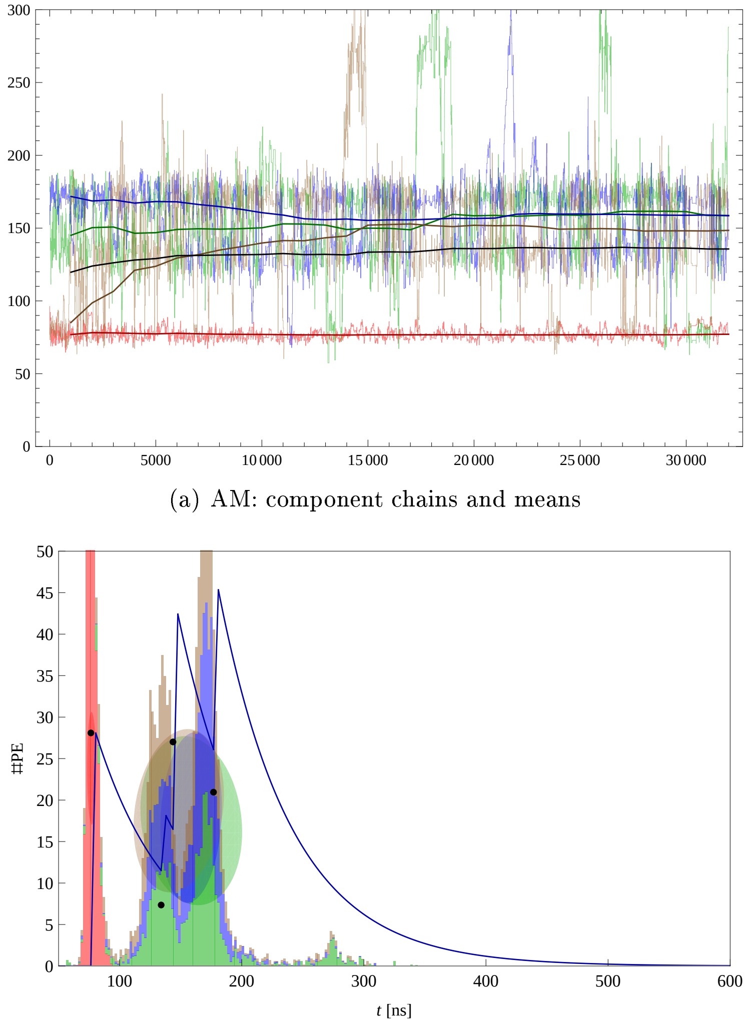
8.2 Relabeling algorithms

In order to favor interpretable unimodal posterior marginals, several methods have been proposed, which we now review. We build upon the review [80] and complete it with recent advances. We also deliberately insist on aspects that will be of importance in our contributions of the next chapters. Sections 8.2.1 to 8.2.4 are devoted to genuine relabeling algorithms, in that they output a relabeled sample, while Sections 8.2.5 and 8.2.6 shortly describe procedures that bypass the label switching problem, each in its own way.

We will always denote by $\pi : \mathbb{R}^d \to \mathbb{R}$ the target distribution of the considered MCMC algorithms, and assume that π is invariant to the permutations contained in $\mathcal{P}$: for any $x \in \mathbb{R}^d$ and any $\sigma \in \mathcal{P}$, $\pi(x) = \pi(\sigma(x))$. Furthermore, all mixture models considered in this section are of fixed and known number of components K.

8.2.1 Imposing an identifiability constraint

Since π is invariant to the action of $\mathcal{P}$, it has $|\mathcal{P}|$ redundant modes that are permuted copies of each other. Note that what we call *mode* in this chapter is a restriction of the target π to an area, from which π could be recovered by applying all permutations in $\mathcal{P}$. We use the term *mode* in a loose acception, since the restriction of π to one



(a) AM: component chains and means

(b) AM: component posteriors

Figure 8.1: The results of AM on an example Poisson tank signal. Panel 8.1(b) shows the parameters of the four muons. The x-coordinates of the black dots are the four times or arrival, while the y-coordinates are the corresponding amplitudes. The blue curve is the ideal PE response. Colored histograms depict the marginal posteriors of the four arrival times. Shaded ellipses are exp(1/2)-level sets of Gaussian distributions: the means are the Bayesian estimates for the arrival time and amplitude of each muon, and the covariance is the marginal posterior covariance of each arrival time/amplitude couple. Panel 8.1(a) shows the four chains of the arrival times (light colors), the running means of all marginal chains (dark colors), and the mean of the running means (black curve). The AM algorithm shows heavy label switching among the three rightmost components.

of these *modes* can well be multimodal; in this case, following the label switching literature, we will speak of *genuine* multimodality. Classical relabeling algorithms work by choosing one of these copies and constrain the sample to remain in this area. The first idea that came up was to add a constraint to the prior that forces the chain to visit a single mode of the posterior, this mode being chosen prior to the experience by the user. In the example of Figure 8.1, it could, for instance, correspond to enforce $t_1 < t_2 < t_3 < t_4$ by multiplying the prior on the vector $(\mathbf{t}, \mathbf{A})$ by $\mathbb{1}_{\mathcal{C}}$ where

$$\mathcal{C} = \{(\mathbf{t}, \mathbf{A}) \ / \ t_1 < t_2 < t_3 < t_4\} \ .$$

A similar method that leads to higher acceptance consists in permuting each candidate point of the MCMC sampler targeting the unconstrained posterior π so that the permuted candidate satisfies the identifiability constraint. These two methods are widely used in practice since they are easy to implement. They are empirically assessed in [60]. In practice, one can run an unconstrained MCMC algorithm targeting the original posterior, apply several constraints to the posterior sample in a *post-processing* step, and select the constraint that yields the best looking marginals.

There are two important downsides to identifiability constraints. First, the choice of the constraint is left to the user, and a bad choice might well not respect the topology of the posterior [94] and lead to artificial biases. In our example of Figure 8.1, if the signal contains two very close arrival times $t_1 \approx t_2$ with similar amplitudes $A_1 \approx A_2$, imposing $t_1 < t_2$ in the sample will artificially censor the chains and yield estimates of t_1 and t_2 that are further away from each other than expected. Second, simply permuting the candidate points of an MCMC sampler without any repercussion on the acceptance ratio does not lead to a balanced algorithm, thus making the target distribution of the chain unclear. We have not seen this last point discussed in the literature, and we will discuss it again in Chapters 9 and 10.

8.2.2 Pivotal relabeling

Since a bad choice of identifiability constraint can lead to ill-shaped restrictions of the posterior, the authors of [94] have proposed to post-process the sample $\mathbf{x} = (x_1, \dots, x_N)$ targeting π as follows: first fix $x_{\mathrm{MAP}} = \arg\max_i \pi(x_i)$ to be an estimate of the maximum of the target, and then replace each sample x_i by $\sigma_i(x_i)$ where

$$\sigma_i = \arg\min_{\sigma \in \mathcal{P}} \|\sigma(x_i) - x_{\mathrm{MAP}}\| \ ,$$

Loosely said, the selected mode of the posterior is then the most circular possible, centered at x_{MAP}. This method is a way to automatically choose an identifiability constraint and apply it, and thus inherits the relatively small computational cost of this approach. However, in case of genuine multimodality of the target, that is, multimodality within one of the modes duplicated by the permutation invariance, the selected region around the target maximum might still be a poor choice of constraint. Intuitively, the mean of the target over the selected region would be a better choice of pivot, see the follow-up in Section 8.2.4.

8.2.3 Constraining the allocation

Consider performing inference on the parameters of a mixture model. If one has a realization $(x_i, z_i)_{i=1,...,N}$ of a chain targeting the complete posterior over both the mixture parameters x and the allocation vector $z \in \{1, \dots, K\}^N$, the authors of [101] propose to put a constraint on the allocation variable z by relabeling each sample (x_i, z_i) so that the relabeled allocation belongs to a previously fixed set $\mathcal{Z}_0$. While the experiments demonstrate quite a general set of constraints available, the choice of the set $\mathcal{Z}_0$ as described in [101] is rather sophisticated and, in the end, user-dependent. Furthermore, there is no theoretical guarantee that the output relabeled sample is concentrated exactly on a single symmetric mode of the posterior.

Let us add that suggestions are made in [101] to choose $\mathcal{Z}_0$ that are inspired by the pivotal relabeling algorithm described in Section 8.2.2, while speed of convergence is addressed in [102].

8.2.4 Learning the constraint

An important reference in the relabeling literature is [123]. Its main contribution is a post-processing relabeling MCMC algorithm: once a realization $\mathbf{x} = (x_1, ..., x_N)$ of a chain targeting the unconstrained posterior π has been drawn, one performs inference tasks as usual but with the *relabeled sample*, defined as

$$\boldsymbol{\sigma}(\mathbf{x}) = \{\sigma_1(x_1), \dots, \sigma_N(x_N)\} \ ,$$

where

$$\boldsymbol{\sigma} = (\sigma_1, \dots, \sigma_N) = \underset{\mathcal{P}\times\cdots\times\mathcal{P}}{\arg\min}\, L(\mathbf{x}, \boldsymbol{\sigma}) \ ,$$

and L is a user-defined cost function. Explicit choices for L are given in [123], among which

$$L(\mathbf{x}, \boldsymbol{\sigma}) = \prod_{i=1}^{N} \mathcal{N}(\sigma_i(x_i) | \mu_N^{\boldsymbol{\sigma}}, \Sigma_N^{\boldsymbol{\sigma}}) \ , \tag{8.2.1}$$

where $\mathcal{N}(\cdot|\mu, \Sigma)$ denotes the Gaussian pdf with mean μ and covariance matrix Σ, and

$$\mu_N^{\boldsymbol{\sigma}} = \frac{1}{N} \sum_{i=1}^{N} \sigma_i(x_i) \ ,$$

$$\Sigma_N^{\boldsymbol{\sigma}} = \frac{1}{N} \sum_{i=1}^{N} (\sigma_i(x_i) - \mu_N^{\boldsymbol{\sigma}})(\sigma_i(x_i) - \mu_N^{\boldsymbol{\sigma}})^T \ .$$

The Gaussian cost function (8.2.1) translates the idea that one wants a relabeled sample to be the most Gaussian possible *among its permutations* $\boldsymbol{\sigma}(\mathbf{x}), \boldsymbol{\sigma} \in \mathcal{P}^N$, in order for $\boldsymbol{\sigma}(\mathbf{x})$ to look as unimodal as possible. Loosely speaking, this approach is a sophisticated version of the pivotal reordering described in Section 8.2.2, replacing the sample maximum of the posterior by a relabeled sample mean and replacing the Euclidean distance by a Mahalanobis distance dictated by the relabeled sample covariance.

This post-processing approach can be seen as a way to automatically learn a reasonable identifiability constraint. However, it is particularly costly since it involves a combinatorial optimization over $\mathcal{P}^N$, which is unfeasible in practice: if π is defined on $\mathbb{R}^d$ and $\mathcal{P}$ is the group $\mathfrak{S}_d$ formed by the permutations of d elements, $\mathcal{P}^N$ has cardinal $(d!)^N$. Approximate optimization algorithms will further render the relabeled target difficult to identify. Cost functions that yield easier optimization tasks at the price of interpretability are advocated in [123], and a similar approach can be found in [47].

An online approach to the optimization of the Gaussian cost function was proposed in [36], and is actually older than the post-processing approach of [123]. We have delayed its presentation until now for the sake of clarity. The approach of [36], henceforth referred to as Celeux's algorithm, is to use the online estimates μ_i and Σ_i of the mean and covariance of the relabeled sample to relabel each candidate point of the MCMC algorithm: after drawing a candidate $\tilde{x}$ at iteration i, $\tilde{x}$ is replaced by $\sigma(\tilde{x})$, where

$$\sigma = \arg\min_{\sigma\in\mathcal{P}} \mathcal{N}\big(\sigma(\tilde{x})|\mu_i, \Sigma_i\big) \ , \tag{8.2.2}$$

before being accepted or not. This way, the sample is relabeled online, and the final cost is in $Nd!$. Furthermore, Celeux's original algorithm only considers diagonal matrices Σ_i, zeroing all cross-covariance terms, thus making the optimization task (8.2.2) computationally cheaper but sacrificing the genericity of the learned identifiability constraint, as we will demonstrate in Chapter 9. Another downside is again the difficulty to interpret the distribution of the relabeled sample, especially if we add the fact that the relabeling should be taken into account in the acceptance ratio of any online relabeling algorithm in order to yield a well-defined MCMC algorithm, while it is not in [36]. A related open question is that of the convergence of the estimates μ_i and Σ_i. We will address these questions in Chapters 9 and 10.

8.2.5 Probabilistic relabeling strategies

It has been proposed in [79] to consider that each realization $x_i, i = 1 \dots N$ of the chain is associated to an unknown permutation σ_i and to approximate distributions $g_i(\cdot;\bar{x})$ over the permutations attached to each realization x_i, given a pivotal value $\bar{x} \in \mathbb{R}^d$ of the parameters of interest. Estimation of a function $h(x)$ is then performed by averaging

$$\widehat{h(x)} = \frac{1}{N}\sum_{i=1}^{N}\sum_{\sigma\in\mathcal{P}} h(\sigma(x_i))g_i(\sigma;\bar{x}) \ .$$

The construction of $\bar{x}$ and g_i is a difficult task. Post-processing suggestions are proposed in [79]. They work by

- running the sampler for a few steps, visually checking that the chain has not switched,
- taking $\bar{x}$ to be the sample mean of that small sample,
- estimate g_i with a method similar to the expectation-maximization (EM; [49]) algorithm.

In [121], EM and stochastic EM variants are described, which explicitly consider the permutation attached to each sample as a hidden variable.

8.2.6 Permutation invariant loss functions

Depending on the problem considered, there might be other ways to bypass the label switching problem, as described in [37]. Let $\mathcal{L} : \mathbb{R}^d \times \mathcal{A} \rightarrow \mathbb{R}$ be a loss function that is invariant to relabeling

$$\mathcal{L}(x, a) = \mathcal{L}(\sigma(x), a), \ \forall \sigma \in \mathcal{P}, \ \forall a \in \mathcal{A} ,$$

where $\mathcal{A}$ is an action space that corresponds to the task to perform. If the aim of the inference task can be described by the minimization problem

$$\min_a \mathbb{E}_\pi \mathcal{L}(x, a) ,$$

then an approximate solution can be obtained by minimizing

$$\sum_{i=1}^{N} \mathcal{L}(x_i, \cdot) ,$$

for which stochastic optimization algorithms can be used.

As an example, consider performing Bayesian inference on the parameters $x \in \mathcal{M} \subset \mathbb{R}^d$ of a mixture distribution. $\mathcal{A}$ can then be taken to be $\mathcal{M}$, and a choice of invariant loss function $\mathcal{L}(x, a)$ is the squared distance between the mixture distributions corresponding respectively to parameters x and a. Other examples of loss functions and algorithms can be found in [37] and [76].

This approach has two advantages: label switching is simply not a problem anymore, and its Bayesian decision theoretic framework is elegant. However, it still necessitates a good stochastic optimization procedure and the existence of a suitable loss function for the problem considered, which is not the case in general.

8.3 Conclusion and reading map

Solving or bypassing the label switching problem has been given much attention in the statistics community, and different methods have been proposed, each with its advantages and drawbacks. Many solutions exist to output a relabeled sample that targets a constrained target, which hopefully matches a single out of the many symmetric modes of the target. The two main downsides to existing methods are that

- they often require human intervention to set up free parameters,
- there is in general no guarantee on the convergence of the relabeled sample, either because the MCMC algorithm is not balanced, or because local and/or approximate optimization algorithms are used.

These points become big issues when performing inference in a large model with permutation invariance where human intervention is not possible, and no loss function is available to apply the approach of Section 8.2.6. Inference in a large-scale generative model of Auger, part of which is presented in Section 7.2, is such a task. In Chapter 9, we propose AMOR, an adaptive MCMC algorithm with an online relabeling mechanism that builds on the approaches of Section 8.2.4. As an adaptive MCMC algorithm, AMOR automatically tunes its proposal distribution. In Chapter 10, we identify the target of AMOR, which does not depend on the user or implementation parameters. We also prove a convergence result on AMOR that answers the question of the convergence of the empirical mean of the chain. Our results are easily applicable to a modified version of Celeux's algorithm, and an online version of the algorithm in [123] also presented in Section 8.2.4.

Chapter 9

AMOR: adaptive Metropolis with online relabeling

Contents

In Chapter 8, we described various approaches to deal with label switching. In the current chapter, we present AMOR, a novel doubly-adaptive MCMC algorithm with online relabeling, which learns both its target and its proposal on the fly, tying the two adaptations in an efficient manner. We first published AMOR in [14]. In [16], we applied it to the Auger tank signal model of Section 7.2. Recently, we submitted in [13] the illustrative example of Section 9.2.2 and the proof in Chapter 10. This is joint work with Olivier Cappé, Gersende Fort (both at CNRS & Télécom ParisTech), and my advisor Balázs Kégl.

9.1 Introduction

In this chapter, we address the label switching problem in the generic case where no useful external information on the target is known. This corresponds, for instance, to a posterior distribution when neither the likelihood is assumed to have a specific form, nor the prior is chosen to have conjugacy properties, which forbids the use of Gibbs sampling or other specialized sampling strategies. We assume, however, that the target is known to be invariant under some permutations of the parameters. This framework is typical, for instance, in experimental physics applications where the likelihood computation is commonly deferred to a *black-box* numerical code. In those cases, one cannot assume anything about the structure of the posterior or its conditional distributions, except that they should be invariant to some permutations of the parameters. We also restrict ourselves to the case where the dimension of the model is finite and known so the parameters of the model are $\mathbb{R}^d$-valued for some fixed d.

Adaptive MCMC algorithms can self-calibrate their internal parameters along the iterations in order to reach decent performance without (or with almost no) knowledge about the target distribution, thus automatizing the grueling step of tuning the proposals. Adaptive MCMC has been an active field of research in the last ten years, following the pioneering contribution of [69] — see [8] as well as the other papers in the same special issue of *Statistics and Computing*, along with [9, 7, 114]. Adaptive Metropolis (hereafter AM; [69]) and its variants aim at identifying the unknown covariance structure of the target distribution along the run of a random walk Metropolis-Hastings algorithm with a multivariate Gaussian proposal. The rationale behind this approach is based on scaling results which suggest that, when d tends to $+\infty$, the chain correlation is minimized when the covariance matrix used in the proposal distribution matches, up to a constant that depends on the dimension, the covariance matrix of the target, for a large class of unimodal target distributions with independent marginals [111, 112]. AM thus progressively adapts, using a stochastic approximation scheme, the covariance of the proposal distribution to the estimated covariance of the target.

It has been empirically observed in [14], and we provide further evidence of this fact below in Section 9.2.2, that the efficiency of AM can be greatly impaired when label switching occurs. The reason for such a difficulty is obvious: if label switching occurs, the estimated covariance matrix no longer corresponds to the local shape of the modes of the posterior and so the exploration can be far from optimal. In Section 9.2.2, we also provide some empirical evidence that off-the-shelf solutions to the label switching problem, such as imposing identifiability constraints or post-processing the simulated sample, are not fully satisfactory. A key difficulty here is that most of the approaches proposed in the literature are based on post-processing of the simulated trajectories *after* the MCMC algorithm has been fully run [123, 79, 80, 101, 121, 94, 116]. Unfortunately, in the case of adaptive MCMC, post-processing cannot solve the improper exploration issue described above. On the other hand, online relabeling algorithms [109, 37] often require manual tuning based on, for example, prior knowledge on the location of the redundant modes of the target. Without such manual tuning they often yield poor samplers, as we will show it in Section 9.2.2.

In this chapter, we describe an adaptive Metropolis algorithm with online relabeling, called AMOR, building on the approaches reviewed in Section 8.2.4. Our idea is to nest relabeling steps within the MCMC algorithm based on the estimation of a *single covariance matrix* that is used *both* for adapting the covariance of the proposal distribution used in the Metropolis algorithm step *and* for online relabeling. Unlike [36], the AMOR algorithm also corrects for the relabelings using a modified acceptance ratio.

In Section 9.2.2, we provide empirical evidence that the coupling established in AMOR between the criterion used for relabeling and the estimation of the covariance of the local modes of the posterior is beneficial to avoid the distortion of the marginal distributions. Furthermore, the example considered in Section 9.2.2 also demonstrates that the AMOR algorithm samples from non-trivial identifiable restrictions of the posterior distribution, that is, truncations of the posterior on regions where the posterior marginals are distinct but from which the complete posterior can be recovered by permutation. The study of the convergence of AMOR in Section 10.1 reveals an interesting connection with the problem of optimal probabilistic quantization [65] which was implicit in earlier works on label switching. It was observed previously by [100] that some adjustments to the usual theory of stochastic approximation are necessary to analyze online optimal quantification due to the presence of points where the mean field of the algorithm is not differentiable. To circumvent this difficulty, we introduce the stable AMOR algorithm, a novel variant of the AMOR algorithm that avoids these problematic points of the parameter space. Finally, we establish a consistency result for the stable AMOR algorithm, showing that it indeed asymptotically provides samples distributed under a suitably defined restriction of the posterior distribution in which the parameters are marginally identifiable.

The chapter is organized as follows. In Section 9.2, we describe the AMOR algorithm. In Section 9.2.2 we compare AMOR with alternative approaches on an illustrative example. We defer the theoretical analysis of AMOR to Chapter 10.

9.2 The AMOR algorithm

In this section, we briefly review the AMOR algorithm and illustrate its performance on an artificial example.

9.2.1 The algorithm

Let π be a density with respect to (w.r.t.) the Lebesgue measure on $\mathbb{R}^d$ which is invariant to the action of a group $\mathcal{P}$ of matrices, that is,

$$\forall x \in \mathbb{R}^d,\ \forall P \in \mathcal{P},\ \pi(x) = \pi(Px) .$$

Denote by $\mathcal{C}_d^+$ the set of $d \times d$ real positive definite matrices. For $\mu \in \mathbb{R}^d$ and $\Sigma \in \mathcal{C}_d^+$, define $L_\theta : \mathbb{R}^d \to \mathbb{R}_+$ by

$$L_\theta(x) = (x-\mu)^T \Sigma^{-1} (x-\mu) , \tag{9.2.1}$$

and let $\mathcal{N}(\cdot|\mu, \Sigma)$ denote the Gaussian density with mean μ and covariance matrix Σ. The pseudocode of the AMOR algorithm of [14] is given in Figure 9.1.

```
AMOR$\big(\pi(.), X_0, T, \theta_0 = (\mu_0, \Sigma_0), c, (\gamma_t)_{t\geq 0}\big)$
 1  $\mathcal{S} \leftarrow \emptyset$
 2  for $t \leftarrow 1$ to $T$
 3    $\Sigma \leftarrow c\Sigma_{t-1}$  ▷ scaled adaptive covariance
 4    $\tilde{X} \sim \mathcal{N}\big(\cdot\,|X_{t-1}, \Sigma\big)$  ▷ initial proposal
 5    $\tilde{P} \sim \arg\min_{P\in\mathcal{P}} L_{\theta_{t-1}}\big(P\tilde{X}\big)$  ▷ pick an optimal permutation
 6    $\tilde{X} \leftarrow \tilde{P}\tilde{X}$  ▷ relabel the initial proposal
 7    if $\dfrac{\pi(\tilde{X})\sum_P \mathcal{N}\big(PX_{t-1}|\tilde{X}, \Sigma\big)}{\pi(X_{t-1})\sum_P \mathcal{N}\big(P\tilde{X}|X_{t-1}, \Sigma\big)} > \mathcal{U}[0,1]$ then
 8      $X_t \leftarrow \tilde{X}$  ▷ accept
 9    else
10      $X_t \leftarrow X_{t-1}$  ▷ reject
11    $\mathcal{S} \leftarrow \mathcal{S} \cup \{X_t\}$  ▷ update the posterior sample
12    $\mu_t \leftarrow \mu_{t-1} + \gamma_t\big(X_t - \mu_{t-1}\big)$
13    $\Sigma_t \leftarrow \Sigma_{t-1} + \gamma_t\big((X_t - \mu_{t-1})(X_t - \mu_{t-1})^\intercal - \Sigma_{t-1}\big)$
14    $\theta_t \leftarrow (\mu_t, \Sigma_t)$
15  return $\mathcal{S}$
```

Figure 9.1: The pseudocode of the AMOR algorithm. The relabeling steps that make AMOR differ from the adaptive Metropolis of [69] are in blue. Steps 6 is a uniform draw over the arg min. See text for a step-by-step description of the algorithm.

To explain the proposal mechanism of AMOR, let μ_{t-1} and Σ_{t-1} denote the sample mean and the sample covariance matrix, respectively, at the end of iteration $t-1$, and let $\theta_{t-1} = (\mu_{t-1}, \Sigma_{t-1})$. Let $\mathcal{S}$ be the MCMC sample. At iteration t, a point $\tilde{X}$ is first drawn from a Gaussian centered at the previous state X_{t-1} and with covariance $c\Sigma_{t-1}$, where c implements the optimal scaling results in [111, 112] discussed in Section 9.1 (Steps 3 and 4). Then in Steps 5 and 6, $\tilde{X}$ is replaced by $\tilde{P}\tilde{X}$, where $\tilde{P}$ is a uniform draw over permutations $\arg\min_P L_{\theta_{t-1}}(P\tilde{X})$ that minimize the relabeling criterion (9.2.1).[1] This relabeling step makes the augmented sample $S \cup \{\tilde{P}\tilde{X}\}$ look as Gaussian as possible. Formally, it can be seen as a projection onto the Voronoi cell $V_{\theta_{t-1}}$, where

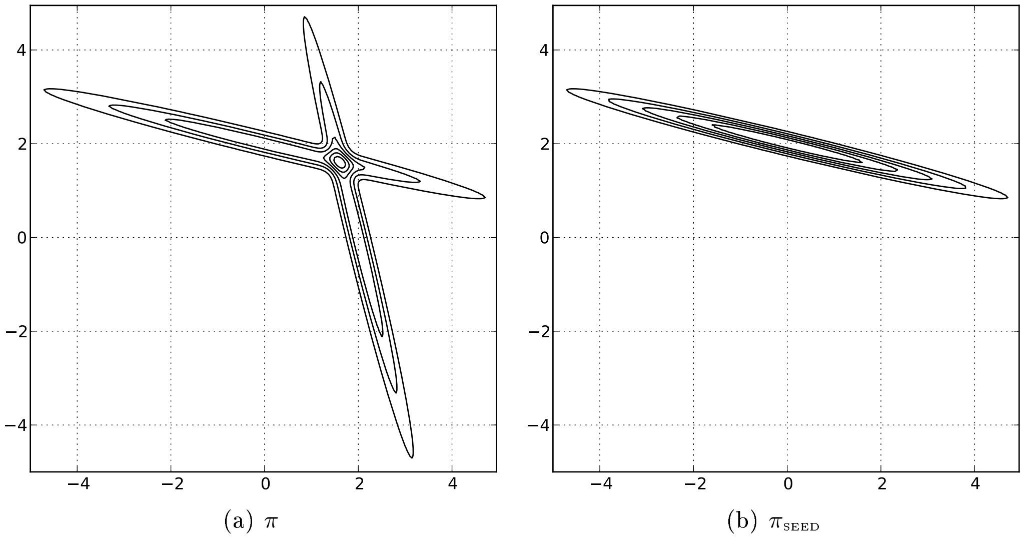
$$V_\theta \ = \{x \in \mathbb{X} \ / \ L_\theta(x) \leq L_\theta(Px), \ \forall P \in \mathcal{P}\} \ . \tag{9.2.2}$$

Then, in Steps 7 to 10, the candidate $\tilde{P}\tilde{X}$ is accepted or rejected according to the usual Metropolis-Hastings rule. Finally, the sample mean and covariance are

[1] Step 5 usually boils down to selecting the permutation $\tilde{P}$ that minimizes $L_{\theta_{t-1}}$. In case of ties, however, $\tilde{P}$ should be drawn uniformly over the set on which the minimum is achieved.

adapted according to a stochastic approximation scheme in Steps 12 to 14 and so (γ_t) is a sequence of nonnegative steps, usually set according to a polynomial decay $\gamma_t \sim t^{-\beta}, \beta \in (1/2, 1]$.

AMOR is a doubly adaptive MCMC algorithm since it is adaptive both in its *proposal* and *relabeling* mechanisms. This means that, besides the proposal distribution, its target also changes with the number of iterations. In Section 10.1 we prove that, at each iteration t, AMOR implements a random walk Metropolis-Hastings kernel with stationary distribution $\pi_\theta \ \propto \pi \, \mathbb{1}_{V_\theta}$.



(a) π

(b) π_{SEED}

Figure 9.2: Panel 9.2(a) shows an example target distribution π, obtained by symmetrizing the Gaussian π_{SEED} shown in Panel 9.2(b).

9.2.2 An illustrative example

In this section, we consider an artificial target aimed at illustrating the gap in performance between the AMOR algorithm and other common approaches to the label switching problem, especially when used within an adaptive MCMC algorithm. Consider the two-dimensional pdf π depicted in Figure 9.2(a), which satisfies $\pi(x) = \pi(Px)$ for $P \in \mathcal{P}$, where

$$\mathcal{P} = \left\{ \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \right\} .$$

The density π is a mixture of two densities with equal weights obtained by superposing the Gaussian pdf π_{SEED} represented in Figure 9.2(b) with a symmetrized version of itself. This artificial target does not correspond to the posterior distribution in an actual inference problem[2]. Nevertheless, it is relevant because it is permutation

[2] In particular, although π itself is a mixture, it is not the posterior distribution of the parameters of any specific mixture model.

invariant and the desired solution of the label switching problem is well-defined: we know that, under suitable relabeling, we can obtain univariate near-Gaussian marginals for both coordinates by recovering the marginals of the two-dimensional Gaussian π_{SEED} in Figure 9.2(b). In spite of its simplicity, this example is challenging because the two marginals of π_{SEED} have similar means (0 and 2) but very different variances (16 and 1), and so they are hard to separate (the correlation between the two coordinates is -0.975).

Given the modest dimension of the problem, we fix the number of iterations to 20 000, of which 4 000 are discarded as burn-in. Unless stated otherwise, all proposals are tuned to reach 50% acceptance of the corresponding MCMC algorithm, which is roughly optimal (at least for unimodal Gaussian densities).

The results of directly applying AM (with no relabeling) are shown in Figure 9.3. The marginal posteriors are sampled quite well (Figures 9.3(c) and 9.3(d)) and the covariance of the joint sample (indicated by a thick ellipse Figure 9.3(a)) is almost symmetric. This is not surprising: the joint distribution, although severely non-Gaussian, is unimodal, and AM has enough time to explore both the original seed π_{SEED} and its symmetric version by frequent label switching. On the other hand, the covariance of the joint distribution π (Figure 9.2(a)) is broader than the covariance of the seed π_{SEED} (Figure 9.2(b)). This results in poor adaptive proposals and slow mixing as indicated by the slight differences between the marginals and the sample marginals, and by autocorrelation function of the first component of the sample in Figure 9.3(b). The reference (dashed line) is the autocorrelation function of an MCMC chain with optimal covariance (proportional to the covariance of the target) targeting the single Gaussian π_{SEED} (Figure 9.2(b)).

We now consider a modified version of AM with online relabeling obtained by simply ordering the variables, meaning that after each proposal $x = (x_1, x_2)$, the components of the proposed point are permuted so that $x_1 \leq x_2$. This strategy is known as *imposing an identifiability constraint*. It is known to perform badly when the constraint does not respect the topology of the target [94]. The results of this approach on our illustrative example are shown in Figure 9.4. The unshaded triangle in Figure 9.4 shows that this time the sample is restricted to a subregion of $\mathbb{R}^2$ where the components are identifiable. Unfortunately, marginals of π restricted to the unshaded triangle in Figures 9.4(c) and 9.4(d) are even more highly skewed than the marginals of the full joint distribution π. Thus, sampling from the restricted distribution π' is not easier than before indicated by the only slightly improved autocorrelation function in Figure 9.4(b).

Applying the ordering constraint *after* the full sample has been drawn with AM leads to similar results as shown in Figure 9.5. This shows that the problem lies with the relabeling criterion rather then with the online nature of the relabeling procedure.

Next, we consider the approach introduced by Celeux in [36]. Celeux's algorithm builds on a non-adaptive isotropic random-walk Metropolis, where online relabeling

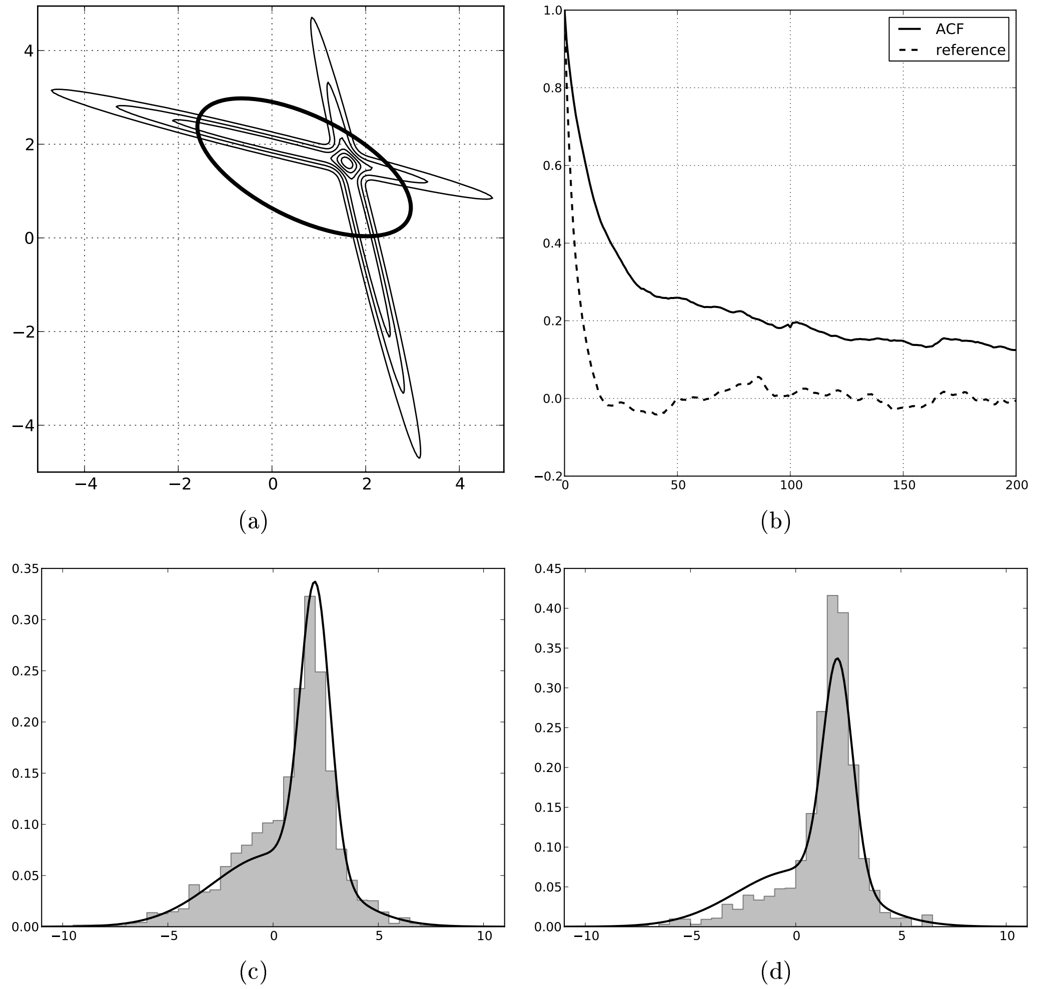
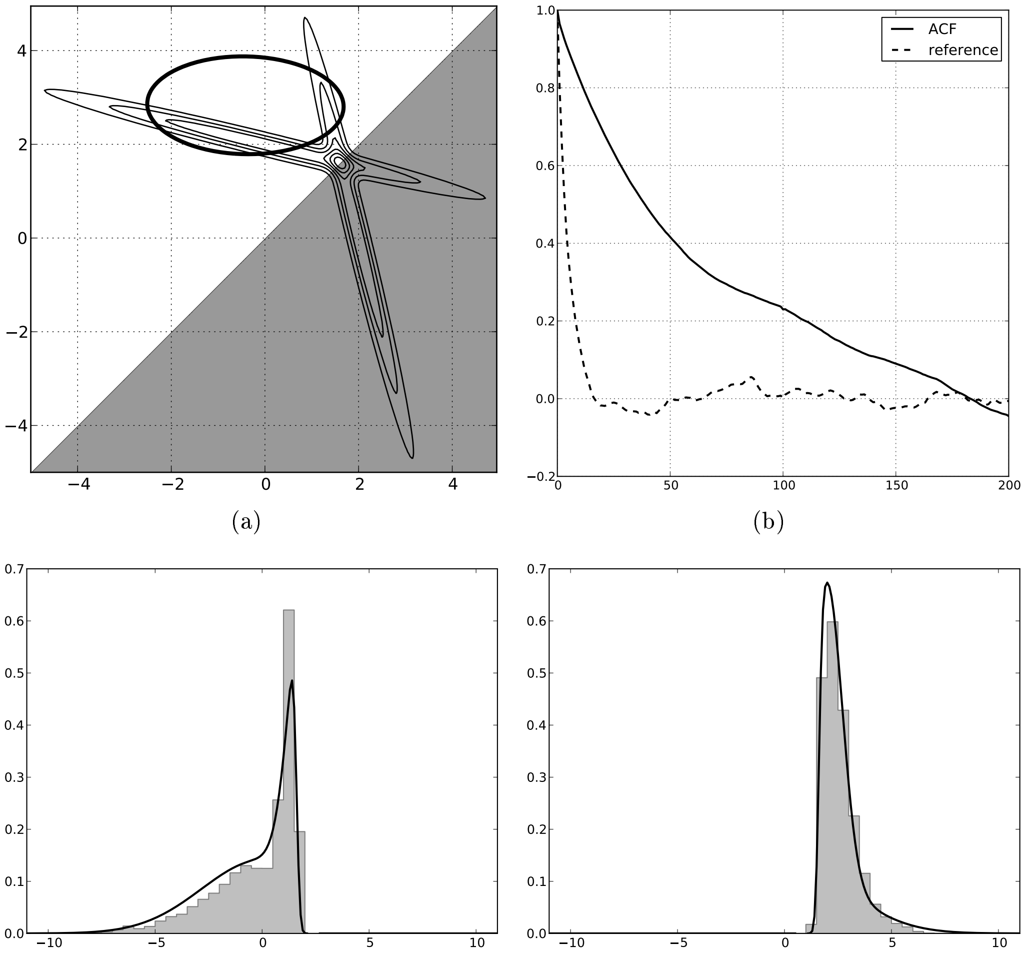


Figure 9.3: Results of vanilla AM on the two-dimensional target π of Figure 9.2. The rest of the caption is the same for Figures 9.4 to 9.7. On Panel 9.3(a), level lines of π are depicted in thin black lines; a thick ellipse centered at the empirical mean μ_T of the sample $\mathcal{S}$ indicates the set $\{x : (x - \mu_T)^T \Sigma_T^{-1}(x - \mu_T) = 1\}$, where Σ_T is the sample covariance. When appropriate, the region of the space selected by (the last iteration of) the algorithm corresponds to the unshaded background while the region not selected is shaded. On Panel 9.3(b), the autocorrelation function (ACF) of the first component of $\mathcal{S}$ is plotted as a solid line. The dashed line indicates the ACF obtained when sampling from the seed Gaussian π_{SEED} of Figure 9.2(b) using a random walk Metropolis algorithm with an optimally tuned covariance matrix. Panels 9.3(c) and 9.3(d) display the histograms of the two marginal samples. The solid curves are the marginals of π in this figure. In Figures 9.4 to 9.7, they are the marginals of π restricted to the unshaded region selected by the algorithms.

is performed in the following way: when a point $x = (x^{(1)}, x^{(2)})$ is proposed at time



(a) (b)

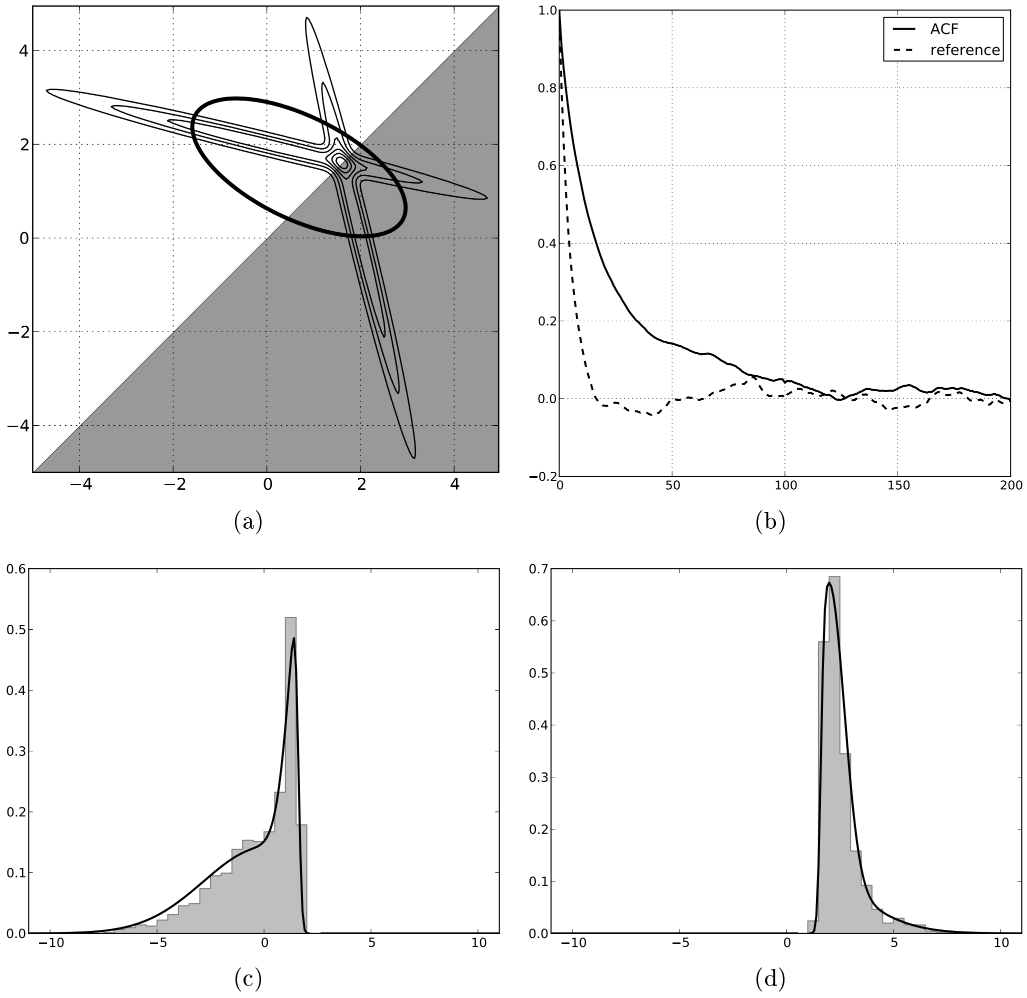
(c) (d)

Figure 9.4: Results of AM with online ordering constraint. For details about the plots, see the caption of Figure 9.3.

t, it is relabeled by

$$x \leftarrow \arg\min \left\{ \begin{pmatrix} x^{(1)} - \mu_t^{(1)} \\ x^{(2)} - \mu_t^{(2)} \end{pmatrix}^T D_t^{-1} \begin{pmatrix} x^{(1)} - \mu_t^{(1)} \\ x^{(2)} - \mu_t^{(2)} \end{pmatrix}, \begin{pmatrix} x^{(2)} - \mu_t^{(1)} \\ x^{(1)} - \mu_t^{(2)} \end{pmatrix}^T D_t^{-1} \begin{pmatrix} x^{(2)} - \mu_t^{(1)} \\ x^{(1)} - \mu_t^{(2)} \end{pmatrix} \right\} . \tag{9.2.3}$$

where $\mu_t = (\mu_t^{(1)}, \mu_t^{(2)})$ is the empirical mean of the current sample $x_{1:t} = x_1, \ldots, x_t$ and D_t is the diagonal matrix containing the empirical variances of the coordinates of $x_{1:t}$ on its diagonal. Formally, this relabeling rule is equivalent to Steps 6 and 7 in Figure 9.1, but with all non-diagonal elements of Σ equal to zero. The results of Celeux's algorithm are shown in Figure 9.6. It is hard to determine precisely the formal target of the algorithm, in particular because the preservation of the detailed balance condition would require incorporating a term into the acceptance ratio to account for the relabeling, which is absent here. It is still possible that the



(a)

(b)

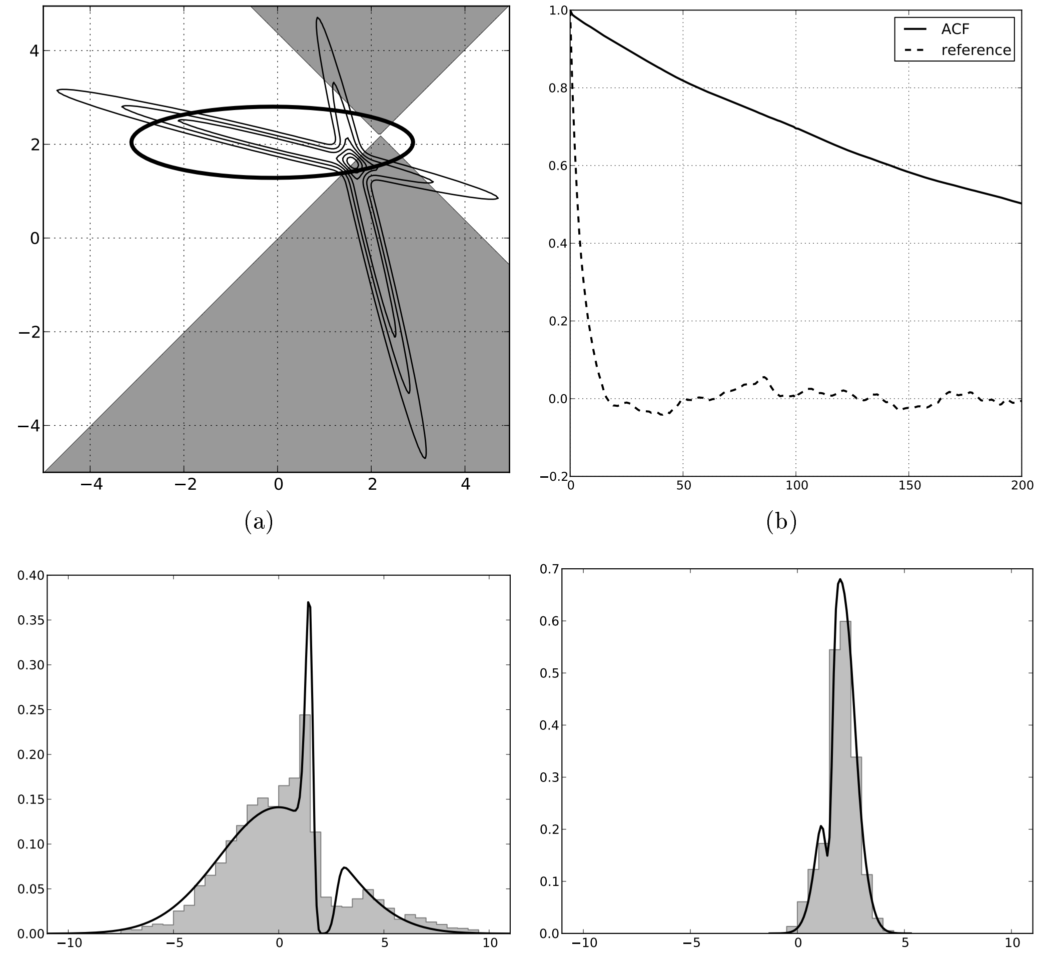
(c)

(d)

Figure 9.5: Results of AM with ordering constraint applied as post-processing. For details about the plots, see the caption of Figure 9.3.

algorithm is *approximately* sampling from the restriction π' of π to this unshaded area in Figure 9.6 (which represents the relabeling rule implemented at the end of the run) in a certain sense. The histograms in Figures 9.6(c) and 9.6(d) *are* similar to the marginals although they are visibly not identical. Certainly, there are no formal guarantees that this should happen. On the other hand, in Section 10.1 we can prove the corresponding claim for the AMOR algorithm.

This relabeling strategy seems to recover π_{SEED} better than the mere ordering of coordinates as suggested by the marginal plots in Figures 9.6(c) and 9.6(d) which are less skewed and now centered at the correct values (0 and 2, respectively). However, using a diagonal covariance D_t also generates some distortion which results in a severely non-Gaussian marginal in Figure 9.6(c). Because of these imperfections and due to the isotropic proposal, the autocorrelation in Figure 9.6(b) indicates, again, a much less efficient sampling than in the case of an optimal Metropolis chain



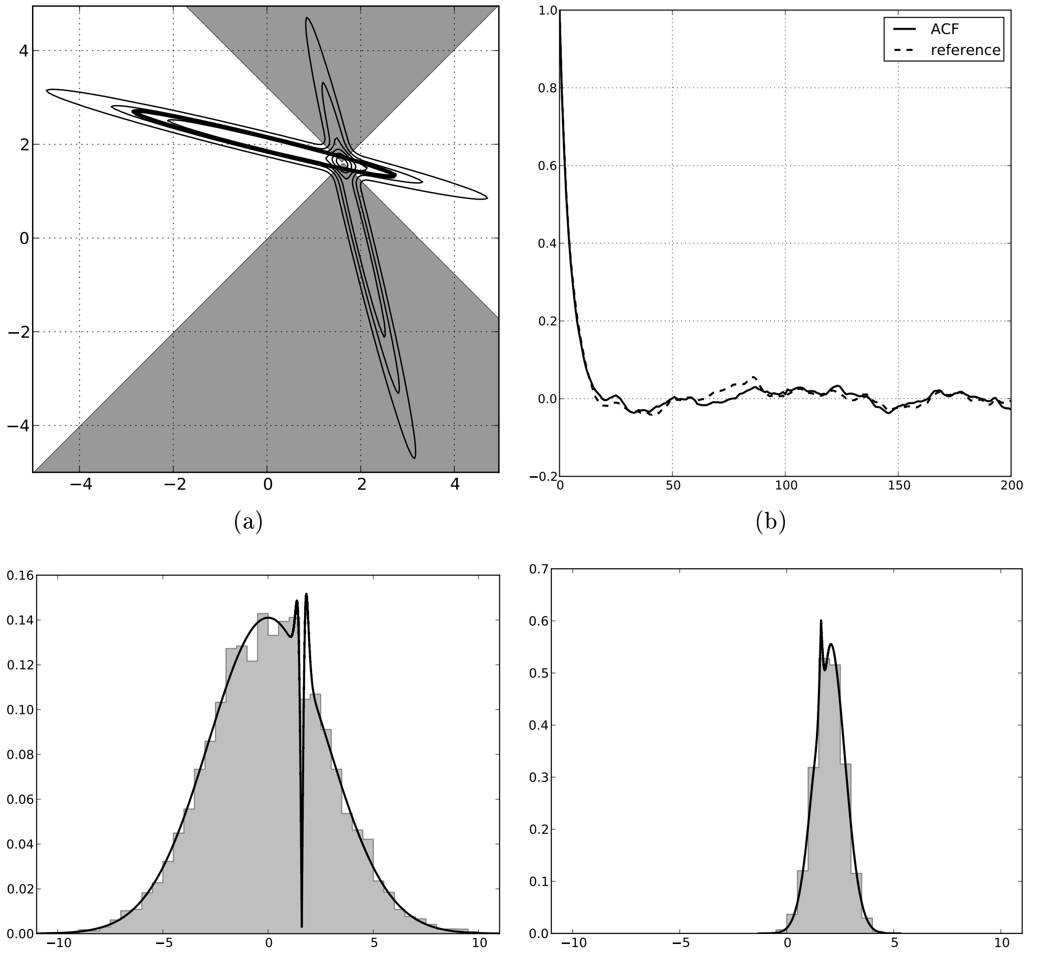
(a) (b)

(c) (d)

Figure 9.6: Results of Celeux's algorithm. For details about the plots, see the caption of Figure 9.3.

targeting π_{SEED}.

The significance of Celeux's algorithm is that its adaptive relabeling rule (9.2.3) makes it possible to resolve the permutation invariance problem in a non-trivial way which appears to be more adapted to the true geometry of the target. It is still not perfect, and, as suggested by [123], one should replace the diagonal covariance matrix in (9.2.3) by the full covariance matrix of the sample. However, [123] explored this idea only as a post-processing approach. A severe difficulty in this context is the computational cost: if T denotes the number of drawn samples and p is the number of permutations to which π is invariant, the required post-processing is a combinatorial problem with p^T possible relabelings. This eventually led [123] to consider a more tractable alternative instead. More importantly in our context, we have seen above (e.g., in Figure 9.3) that running an adaptive MCMC on the full permutation-invariant target may result in a poor mixing performance. To achieve



(a)

(b)

(c)

(d)

Figure 9.7: Results of AMOR. For details about the plots, see the caption of Figure 9.3.

both relevant relabeling and efficient adaptivity, the key idea of AMOR is to link the covariance of the proposal distribution and the covariance used for relabeling, which are proportional to each other in AMOR.

Figure 9.7 displays the results obtained using AMOR on our running example. AMOR does separate $\mathbb{R}^2$ in two regions that respect the topology of the target much more closely than the approaches examined previously. Figure 9.7(a) indicates that the relabeled target is as Gaussian as possible among all partitionings based on a quadratic criterion of the form (9.2.1). The marginals in Figures 9.7(c) and 9.7(d) now look almost Gaussian. They closely match the marginals of both the restricted distribution π' and the seed distribution π_{SEED} in Figure 9.2(b). Furthermore, the autocorrelation function of AMOR (Figure 9.7(b)) is as good as the reference autocorrelation function corresponding to an optimally tuned random walk Metropolis Hastings algorithm targeting the seed Gaussian π_{SEED} in Figure 9.2(b). This perfect adaptation is possible because the sample covariance now matches the covariance of the target restricted to the unshaded region of the plane (Figure 9.7(a)). On this example, the AMOR algorithm thus automatically achieves, *without requiring any manual tuning*, a satisfactory result that cannot be obtained with any of the methods examined previously.

9.3 Application to Gaussian mixtures

As mentioned in Chapter 8, one of the most common use of relabeling happens when performing Bayesian inference in mixture models. We thus benchmark AMOR on two such tasks.

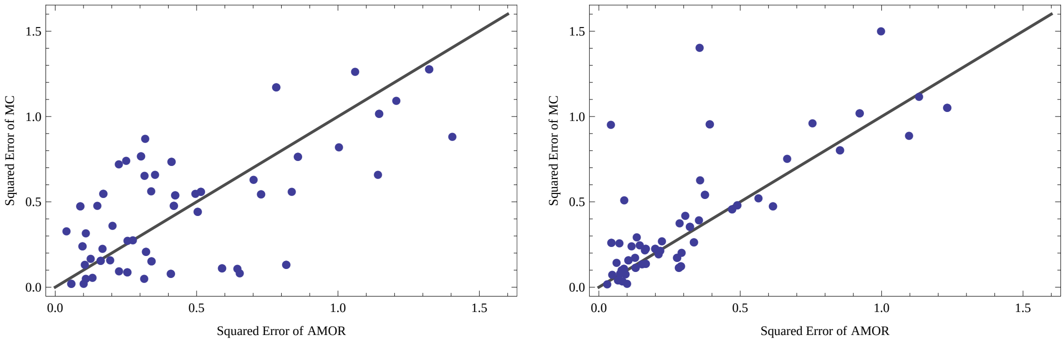
First, we estimate the nine parameters $\psi = (\alpha_i, \mu_i, \sigma_i)_{i=1,2,3}$ of a mixture of three one-dimensional Gaussians $\sum_{i=1}^{3} \alpha_i \mathcal{N}(.|\mu_i, \sigma_i)$, taking wide flat priors over each parameter. Similarly to Section 9.2.2, we compare

- a symmetric random walk Metropolis with an ordering constraint on the three means $\mu_1 \leq \mu_2 \leq \mu_3$ (see Section 8.2.1),
- Celeux's algorithm (see Sections 8.2.4 and 9.2.2),
- Celeux's algorithm with modified acceptance ratio according to (7), henceforth denoted as modified Celeux,
- our AMOR algorithm presented in Section 9.2.

To quantify the performance after T iterations, we first select the permutation of the running posterior mean components $(\hat{\mu}_i^{(T)})_{i=1,2,3}$ which minimizes the sum of the ℓ_2 errors on the three estimates of the means $\mu_i, i = 1, 2, 3$, and we consider the latter sum taken at this best permutation of the posterior mean:

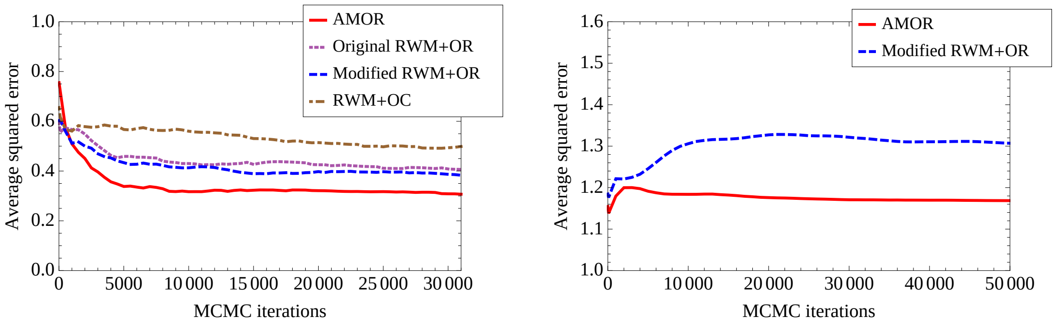
$$S_T = \arg\min_{\tau \in \mathfrak{S}_3} \sum_{i=1}^{3} (\hat{\mu}_{\tau(i)}^{(T)} - \mu_i)^2 \ .$$

We repeated this experiment 100 times on 100 different datasets coming from parameters generated as follows: draw $(\alpha_i) \sim \mathcal{D}(1)$, $\mu_i \sim \mathcal{U}_{(0,1)}$ i.i.d., and $\sigma_i \sim \mathcal{U}_{(0,0.05)}$ i.i.d. This choice of generative distribution ensures a reasonable number of datasets containing overlapping Gaussians, thus provoking switching. Figure 9.9(a) depicts the performance measure S_T averaged over the 100 datasets of this 9D experiment, versus the number T of MCMC iterations. We use this averaging as a way to estimate the expected performance measure on a class of problems given by the generative distribution. AMOR significantly outperforms other approaches as it converges faster on average and to a better solution. As expected, imposing an ordering constraint on the means (RWM+OC on Figure 9.9(a)) reveals a poor strategy leading to artificial additional bias. Note finally that the modification of the acceptance ratio does not increase drastically the performance of the online relabeling algorithm of [36] (“Original RWM+OR” vs. “Modified RWM+OR”), which is not a surprise since the additional factor in the acceptance ratio (Step (7) in Figure 9.1) is often close to 1. Figure 9.8 provides insight into how the two best methods (AMOR and modified Celeux) behaves after $T_1 = 1\,000$ and $T_2 = 30\,000$ iterations, presenting scatterplots of performances $S_{1,000}$ and $S_{30,000}$. Each point corresponds to one of the 100 datasets. Clearly, starting from a rather random distribution of the errors, AMOR took the advantage after 30K iterations.



(a) $S_{1,000}$ in the 9D experiments (b) $S_{30,000}$ in the 9D experiments

Figure 9.8: Experimental comparison of the performance measure S for AMOR (x-axis) versus modified Celeux (y-axis) in the 9D experiment.



(a) 9D experiments (b) 30D experiments

Figure 9.9: Experimental comparison of several relabeling approaches. The plots show the performance measure S_T vs. T averaged over 100 datasets drawn from a common generative model.

To further investigate the comparison between AMOR and MC, we now consider a higher dimensional experiment. This time, the goal is to estimate the three means of a 10-dimensional Gaussian mixture $\sum_{i=1}^{3} 1/3\,\mathcal{N}(.|\mu_i, 0.1I_{10})$. Again, 100 datasets of 100 points each were generated with $\mu_i \sim \mathcal{U}_{(0,1)}$ i.i.d. Again, as seen in Figure 9.9(b), AMOR stabilizes earlier and selects a better region of $\mathbb{R}^{30}$ than modified Celeux, thus illustrating again the interest of combining adaptive selection and proposal mechanisms.

9.4 Application to the Auger tank signal model

The development of an adaptive MCMC algorithm that is able to cope with label switching was originally motivated by the Auger tank signal model of Section 7.2. To demonstrate that we cannot rely on AM being stuck in one of the redundant symmetric modes of the posterior and that relabeling yields improvement, we compare simple AM and AMOR on the task of inferring the 4×2 parameters – four amplitudes $\mathbf{A} = (A_i)_{i=1\dots4}$ and four arrival times $\mathbf{t} = (t_i)_{i=1\dots4}$ – of four vertical centered muons producing a Poisson signal $\mathbf{n}$. As for the initial label switching example in Section 8.1, we do not include the subsequent steps of the model, since at this stage, the likelihood

$$p(\mathbf{n}|\mathbf{t}, \mathbf{A})$$

is already invariant to permutations of the four muons. We ran both AMOR and AM on 1200 simulated tank signals with 20 bins and $N = 4$. To create difficult and realistic situations, we set the prior $p(\mathbf{t})$ to be a product of independent inverse Gamma distributions with parameters 2 and 100. This led to an arrival time distribution with small variance, thus making simulations exhibit a reasonable number of overlapping muons. Examples of such simulated signals are depicted in Figures 9.10(a) and 9.10(b).

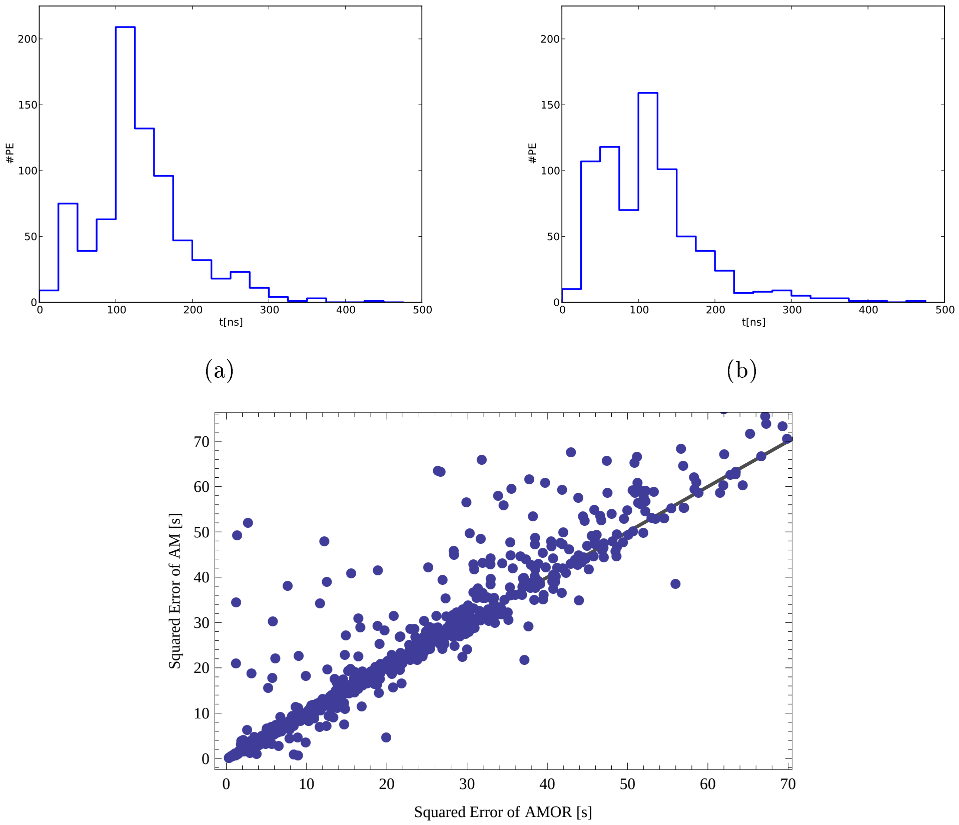
To quantify the performance after T iterations, we first select the permutation of the running posterior mean components $(\hat{t}^{(T)}_{\mu_i})_{i=1,2,3,4}$ which minimizes the sum of the squared errors on the four estimates of the times of arrival $t_{\mu_i}, i = 1, 2, 3, 4,$ and we consider the sum of squared errors taken at this best permutation τ of the posterior mean to compute an error per muon:

$$S_T = \frac{1}{4}\left(\arg\min_{\tau \in \mathfrak{S}_4} \sum_{i=1}^{4} \left(\hat{t}^{(T)}_{\mu_{\tau(i)}} - t_{\mu_i}\right)^2\right)^{1/2} .$$

Figure 9.10(c) shows a scatterplot of S_T after $T = 3 \times 10^6$ iterations. On each signal, both AM and AMOR started with the same initial point, that is all points in Figure 9.10(c) were lying on the diagonal. AMOR clearly outperforms AM on cases where label switching appears, leading to an estimate of the average error per muon of 17.0 ± 0.1ns versus 18.3 ± 0.1ns on these difficult cases.

9.5 Conclusion

We presented AMOR, an adaptive Metropolis-Hastings algorithm that ties the adaptation of its proposal to the online design of its relabeling rule, and demonstrated



(a)

(b)

(c) S_T^{AM} vs. S_T^{AMOR}

Figure 9.10: (a),(b) Examples of generated Auger tank Poisson signals, with 20 bins and 4 muons each. Arrival time distribution is $\mathcal{IG}(2, 100)$. (c) Scatter plots of squared errors of AM vs. AMOR, with $T = 32\,000$ (see text).

AMOR on applications. We now discuss future methodological work on AMOR. The online nature of AMOR makes it cheaper than its post-processing counterpart, but it still requires to sweep over all elements of $\mathcal{P}$ at each iteration. This is prohibitive in problems with large $|\mathcal{P}|$, such as our Auger tank model when the number of muons is large. In future work, we will concentrate on algorithmic modifications to reduce this cost, potentially inspired by *probabilistic* relabeling algorithms [79, 121], while conserving our theoretical results of Section 10.1. Furthermore, we are interested in extending AMOR to trans-dimensional problems, such as mixtures with an unknown number of components. This problem can be addressed by reversible jump MCMC (RJMCMC; [68]), but the latter also suffers from label switching and inferential difficulties, and is not adaptive. We plan to study algorithms that combine RJMCMC and AMOR.

Chapter 10

On the convergence of AMOR

Contents

In Chapter 9, we introduced and demonstrated AMOR, an adaptive Metropolis algorithm with online relabeling. In this chapter, we present our theoretical analysis of AMOR and prove a consistency result that we submitted in [13]. We present here a combination of our papers [13] and [14], which presents the proof of our consistency result in its entirety. This is joint work with Olivier Cappé, Gersende Fort (both at CNRS & Télécom ParisTech), and my advisor Balázs Kégl.

10.1 Main results

AMOR can be cast into the family of adaptive MCMC algorithms such that the updating rule of the design parameter relies on a stochastic approximation scheme. Adaptive MCMC can be described as follows: given a family of transition kernels $(P_\theta)_{\theta\in\Theta}$, the algorithm produces a $\mathbb{X} \times \Theta$-valued process $((X_t, \theta_t))_{t\geq 0}$ such that the conditional distribution of X_t given the past is given by the transition kernel $P_{\theta_{t-1}}$. This algorithm is designed so that when t tends to infinity, the distribution of X_t converges to the invariant distribution of the kernel P_{θ_t}. Convergence of such adaptive procedures was recently analyzed by [113, 59], where sufficient conditions are in terms of the so-called containment condition and the diminishing adaptation property (see [113]). When each transition kernel P_θ possesses its own invariant distribution, a condition on the convergence of these distributions is also required (see [59]).

In the AMOR case, we will show that each transition kernel has its own invariant distribution (see Section 10.1.1). Therefore, as a preliminary step for the convergence of AMOR, the stability and the convergence of the design parameter sequence $(\theta_t)_{t\geq 0}$ have to be established. Sufficient conditions for the convergence of stochastic approximation procedures rely on the existence of a (regular enough) Lyapunov function on Θ, on the behavior of the mean field at the boundary of the parameter set Θ, and on the magnitude of the stepsize sequence $(\gamma_t)_{t\geq 0}$. For AMOR, we were only able to design a Lyapunov function for which some boundaries of Θ are not repulsive (such a preliminary result can be found in [14]). Therefore, we introduced a modified AMOR which differs from the AMOR algorithm in Figure 9.1 through the update rules for (θ_t): a penalty term is added in Steps 12 and 13 of AMOR so that the boundaries of Θ become repulsive for the new algorithm. A stabilization step is also added at the end of the main loop, in order to ensure the sequence $(\theta_t)_{t\geq 0}$ is bounded and it does not get too close to the boundaries of Θ.

The convergence of the stable algorithm is addressed when π is compactly supported.

Assumption 1. *π is a density w.r.t. the Lebesgue measure on $\mathbb{R}^d$, which is bounded, with compact support $\mathbb{X}$, and which is invariant to permutations in the group $\mathcal{P}$:*

$$\forall x \in \mathbb{X}, \forall P \in \mathcal{P}, \pi(Px) = \pi(x) \; .$$

The compacity assumption makes it simpler to analyze the limiting behavior of the algorithm. The proofs can be extended to a more general case by using the same tools as in [59] and [6, section 3]. These technical steps are out of the scope of this thesis.

This chapter is organized as follows. In Section 10.1.1, we first describe the stable AMOR algorithm, and we show that it is an adaptive MCMC algorithm. We then characterize the limiting behavior of the sequence $(\theta_t)_{t\geq 0}$ in Section 10.1.2 and address a strong law of large numbers for the samples $(X_t)_{t\geq 0}$, as well as the ergodicity of the sampler. All proofs are given in Section 10.2.

10.1.1 A stable AMOR algorithm

Set $\mathcal{P}^* = \mathcal{P} \setminus \{\mathrm{Id}\}$ and

$$\Theta = \{(\mu, \Sigma) \in \mathbb{R}^d \times \mathcal{C}_d^+ \; / \; \forall P \in \mathcal{P}^*, \; \Sigma^{-1}\mu \neq P\Sigma^{-1}\mu\} \; . \tag{10.1.1}$$

The set $\mathbb{R}^d \times \mathcal{C}_d^+$ is endowed with the scalar product $\langle (a, A), (b, B) \rangle = a^T b + \mathrm{Trace}(A^T B)$. We will use the same notation $\|.\|$ for the norm induced by this scalar product, for the Euclidean norm on $\mathbb{R}^d$, and for the norm $\|A\| = Tr(A^T A)^{1/2}$ on $d \times d$ real matrices.

Denote by $\mathcal{S}_d$ the set of $d \times d$ symmetric real matrices; and for $P \in \mathcal{P}$, let $U_P = (I-P)^T(I-P)$. Let $\alpha > 0$ be fixed and define $H : \mathbb{X} \times \Theta \to \mathbb{R}^d \times \mathcal{S}_d$ by

$$H(x,\theta) = \big(H_\mu(x,\theta), H_\Sigma(x,\theta)\big) \tag{10.1.2}$$

where

$$\begin{aligned} H_\mu(x,\theta) &= x - \mu - \alpha \sum_{P \in \mathcal{P}^*} \frac{1}{\|(I-P)\Sigma^{-1}\mu\|^4} \; U_P \Sigma^{-1} \mu \;, \\ H_\Sigma(x,\theta) &= (x-\mu)(x-\mu)^T - \Sigma \\ &\quad + \alpha \sum_{P \in \mathcal{P}^*} \frac{1}{\|(I-P)\Sigma^{-1}\mu\|^4} \; \left(\mu\mu^T \Sigma^{-1} U_P + U_P \Sigma^{-1} \mu\mu^T\right) \;. \end{aligned}$$

Finally, for any $\delta > 0$, set

$$\mathcal{K}_\delta = \{(\mu, \Sigma) \in \Theta : \inf_{P \in \mathcal{P}^*} \|(I-P)\Sigma^{-1}\mu\| \geq \delta\} \;. \tag{10.1.3}$$

Let $(\delta_q)_{q \geq 0}$ be a decreasing positive sequence such that $\lim_{q\to\infty} \delta_q = 0$ and $\mathcal{K}_{\delta_0}$ is not empty; choose $\theta_0 = (\mu_0, \Sigma_0) \in \mathcal{K}_{\delta_0}$. Figure 10.1 describes the stable AMOR algorithm in pseudocode.

We now prove that stable AMOR is an adaptive MCMC algorithm. For any $\theta \in \Theta$, define the transition kernel P_θ on $(\mathbb{X}, \mathcal{X})$ by

$$P_\theta(x, A) = \int_{A \cap V_\theta} \alpha_\theta(x,y) q_\theta(x,y) \; dy + \mathbb{1}_A(x) \; \int_{V_\theta} (1 - \alpha_\theta(x,z)) \; q_\theta(x,z) \; dz \;, \tag{10.1.4}$$

where

$$\alpha_\theta(x,y) = 1 \wedge \frac{\pi(y) q_\theta(y,x)}{\pi(x) q_\theta(x,y)} \;, \tag{10.1.5}$$

and

$$q_\theta(x,y) = \sum_{P \in \mathcal{P}} \mathcal{N}(Py | x, c\Sigma) \;. \tag{10.1.6}$$

For $\theta \in \Theta$, define also

$$\pi_\theta = |\mathcal{P}| \mathbb{1}_{V_\theta} \pi \;. \tag{10.1.7}$$

The following proposition shows that $q_\theta(x, \cdot)$ is a density on V_θ, and that the distribution π_θ given by (10.1.7) is invariant for the transition kernel P_θ. It also establishes that Stable AMOR is an adaptive MCMC algorithm: given (X_{t-1}, θ_{t-1}), X_t is obtained by one iteration of a random walk Metropolis-Hastings algorithm with proposal $q_{\theta_{t-1}}$ and invariant distribution $\pi_{\theta_{t-1}}$.

Proposition 1. *1. For any $\theta \in \Theta$ and $x \in \mathbb{X}$, $\int_{V_\theta} \; q_\theta(x,y) \; dy = 1$.*

2. For any $\theta \in \Theta$, $\pi_\theta P_\theta = \pi_\theta$ and for any $x \in V_\theta$, $P_\theta(x, V_\theta) = 1$.

```
STABLEAMOR$(\pi(\cdot), X_0, T, \theta_0 = (\mu_0, \Sigma_0), c, (\gamma_t)_{t\geq 0}, \alpha, (\mathcal{K}_{\delta_q})_{q\geq 0})$

 1  $\mathcal{S} \leftarrow \emptyset$
 2  $\psi \leftarrow 0$   ▷ Projection counter
 3  for $t \leftarrow 1$ to $T$
 4     $\Sigma \leftarrow c\Sigma_{t-1}$ ▷ scaled adaptive covariance
 5     $\tilde{X} \sim \mathcal{N}\big(\cdot\,|X_{t-1}, \Sigma\big)$   ▷ initial proposal
 6     $\tilde{P} \sim \arg\min_{P\in\mathcal{P}} L_{\theta_{t-1}}\big(P\tilde{X}\big)$   ▷ pick an optimal permutation
 7     $\tilde{X} \leftarrow \tilde{P}\tilde{X}$   ▷ relabel the initial proposal
 8     if $\dfrac{\pi(\tilde{X})\sum_P \mathcal{N}\big(PX_{t-1}|\tilde{X}, \Sigma\big)}{\pi(X_{t-1})\sum_P \mathcal{N}\big(P\tilde{X}|X_{t-1}, \Sigma\big)} > \mathcal{U}[0,1]$ then
 9        $X_t \leftarrow \tilde{X}$   ▷ accept
10     else
11        $X_t \leftarrow X_{t-1}$   ▷ reject
12     $\mathcal{S} \leftarrow \mathcal{S} \cup \{X_t\}$   ▷ update posterior sample
13     $\mu_t \leftarrow \mu_{t-1} + \gamma_t H_\mu(X_t, \theta_{t-1})$
14     $\Sigma_t \leftarrow \Sigma_{t-1} + \gamma_t H_\Sigma(X_t, \theta_{t-1})$
15     if $(\mu_t, \Sigma_t) \notin \mathcal{K}_{\delta_\psi}$ then
16        $(\mu_t, \Sigma_t) \leftarrow (\mu_0, \Sigma_0)$   ▷ Project back to $\mathcal{K}_{\delta_0}$
17        $\psi \leftarrow \psi + 1$   ▷ Increment projection counter
18     $\theta_t \leftarrow (\mu_t, \Sigma_t)$
19  return $\mathcal{S}$
```

Figure 10.1: Pseudocode of stable AMOR. Step 6 is a uniform draw over the arg min. There are two modifications w.r.t. AMOR in Figure 9.1, depicted in blue in the current figure. The first one is about the updates of μ and Σ in Steps 13 and 14, with $H(\mu, \Sigma)$ being defined in (10.1.2). The second one is the projection mechanism, Steps 15 to 17. These modifications prevent the new value (μ_t, Σ_t) to jump to the set $\{\theta \in \Theta : \inf_{P\in\mathcal{P}^*} \|(I-P)\Sigma^{-1}\mu\| = 0\}$.

3. *Let $(\theta_t, X_t)_{t\geq 0}$ be given by the stable AMOR in Figure 10.1. Conditionally on $\sigma(X_0, \theta_0, X_1, \theta_1, ..., X_{t-1}, \theta_{t-1})$, the distribution of X_t is $P_{\theta_{t-1}}(X_{t-1}, \cdot)$.*

Note that the proof of Proposition 1 is independent of the update scheme of $(\theta_t)_{t\geq 0}$, which makes the proposition valid for both AMOR in Figure 9.1 and stable AMOR in Figure 10.1.

10.1.2 Convergence of stable AMOR

Let $\mu_{\pi_\theta}, \Sigma_{\pi_\theta}$ be the expectation and covariance matrix of π_θ

$$\mu_{\pi_\theta} = \int x \; \pi_\theta(x) \; dx \; , \tag{10.1.8}$$

$$\Sigma_{\pi_\theta} = \int (x - \mu_{\pi_\theta})(x - \mu_{\pi_\theta})^T \; \pi_\theta(x) \; dx \; . \tag{10.1.9}$$

Define the *mean field* $h : \Theta \to \mathbb{R}^d \times \mathcal{S}_d$ by

$$h(\theta) = \big(h_\mu(\theta), h_\Sigma(\theta)\big) \tag{10.1.10}$$

where

$$\begin{aligned} h_\mu(\theta) &= \mu_{\pi_\theta} - \mu - \alpha \sum_{P \in \mathcal{P}^*} \frac{1}{\|(I-P)\Sigma^{-1}\mu\|^4} \; U_P \Sigma^{-1} \mu \; , \\ h_\Sigma(\theta) &= \Sigma_{\pi_\theta} - \Sigma + (\mu_{\pi_\theta} - \mu)(\mu_{\pi_\theta} - \mu)^T \\ &\quad + \alpha \sum_{P \in \mathcal{P}^*} \frac{1}{\|(I-P)\Sigma^{-1}\mu\|^4} \; \left(\mu\mu^T \Sigma^{-1} U_P + U_P \Sigma^{-1} \mu\mu^T\right) \; . \end{aligned}$$

The key ingredient for the proof of the convergence of the sequence $(\theta_t)_{t \geq 0}$ is the existence of a Lyapunov function w for the mean field h: we prove in Section 10.2 (see Lemma 10) that the function $w : \Theta \to \mathbb{R}_+$, defined by

$$w(\theta) = -\int \log \mathcal{N}(x|\theta) \pi_\theta(x) \; dx + \frac{\alpha}{2} \sum_{P \in \mathcal{P}^*} \frac{1}{\|(I-P)\Sigma^{-1}\mu\|^2} \; , \tag{10.1.11}$$

is continuously differentiable on Θ and satisfies $\langle \nabla w, h \rangle \leq 0$. In addition, $\langle \nabla w(\theta), h(\theta) \rangle = 0$ iff θ is in the set

$$\mathcal{L} = \{\theta \in \Theta : h(\theta) = 0\} = \{\theta \in \Theta : \nabla w(\theta) = 0\} \; . \tag{10.1.12}$$

The convergence of the sequence $(\theta_t)_{t \geq 0}$ is proved by verifying the sufficient conditions for convergence of stochastic approximation for Lyapunov stable dynamics given in [6]. The first step consists in proving that the sequence is bounded with probability one: we prove that, almost surely, the number of projections ψ is finite so that the projection mechanism (lines 15 to 17 in Figure 10.1) never occurs after a (random) number of iterations. We then prove the convergence of the stable sequence. To that goal, following the same lines as in [6], it is assumed

Assumption 2. *Let $\mathcal{L}$ be given by (10.1.12). There exists $M_\star > 0$ such that $\mathcal{L} \subset \{\theta : w(\theta) \leq M_\star\}$, and $w(\mathcal{L})$ has an empty interior.*

Define for $x \in \mathbb{R}^d$ and $A \subset \mathbb{R}^d$, $\mathrm{d}(x, A) = \inf_{a \in A} \|x - a\|$. The following result is proved in Section 10.2.

Theorem 2. *Let $\beta \in (1/2, 1]$ and $\gamma_\star > 0$. Let $(\theta_t)_{t \geq 0}$ be the sequence produced by the stable AMOR algorithm in Figure 10.1 with $\gamma_t \sim \gamma_\star t^{-\beta}$ when $t \to +\infty$. Under Assumptions 1 and 2,*

1. *Almost surely, there exist $M > 0$ and $t_\star > 0$ such that for any $t \geq t_\star$, $\theta_t \in \{\theta \in \Theta : w(\theta) \leq M\}$. In addition, the number of projections is finite almost-surely.*
2. *Almost surely, $\limsup_t \mathrm{d}(\theta_t, \mathcal{L}) \to 0$.*

Theorem 2 states the convergence of $(\theta_t)_{t \geq 0}$ to the set $\mathcal{L}$ of the zeros of h. This is a classical type of result in stochastic approximation theory. Pointwise convergence can be obtained by further assuming that, e.g., the set $\mathcal{L}$ is a union of isolated points. In practice, it is hard to check whether such an assumption is satisfied. However, in our experiments, we always observed pointwise convergence to a limiting value that did not depend on initialization of the algorithm or implementation parameters.

We now state a strong law of large numbers for the samples $(X_t)_{t \geq 0}$, which holds for all paths such that $(\theta_t)_t$ converges to a point $\theta^\star \in \mathcal{L}$.

Theorem 3. *Let $\beta \in (1/2, 1]$, $\gamma_\star > 0$ and $\theta^\star \in \mathcal{L}$. Let $(X_t, \theta_t)_{t \geq 0}$ be the sequence generated by the stable AMOR algorithm in Figure 10.1 with $\gamma_t \sim \gamma_\star t^{-\beta}$ when $t \to +\infty$. Under Assumptions 1 and 2, on the set $\{\lim_t \theta_t = \theta^\star\}$*

$$\lim_{T \to \infty} \frac{1}{T} \sum_{t=1}^{T} f(X_t) = \pi_{\theta^\star}(f) \ ,$$

for any bounded function f.

Finally, Theorem 4 yields the ergodicity of AMOR.

Theorem 4. *Let $\beta \in (1/2, 1]$, $\gamma_\star > 0$, and $\theta^\star \in \mathcal{L}$. Let $(X_t, \theta_t)_{t \geq 0}$ be the sequence generated by Algorithm 10.1 with $\gamma_t \sim \gamma_\star t^{-\beta}$ when $t \to +\infty$. Under Assumptions 1 and 2,*

$$\lim_{t \to \infty} \sup_{\|f\|_\infty \leq 1} \left| \mathbb{E}\big[f(X_t) \mathbb{1}_{\lim_q \theta_q = \theta^\star}\big] - \pi_{\theta^\star}(f)\ \mathbb{P}(\lim_q \theta_q = \theta^\star) \right| = 0 \ .$$

The expression of w provides insight into the links between relabeling and vector quantization [65]. The first term in the RHS of (10.1.11) is similar to a distortion measure in vector quantization as noted in [14], and restated here in Section 10.2 as Lemma 6. It can also be seen as the cross-entropy between π_θ and a Gaussian with parameters θ. The second term in the RHS of (10.1.11) is similar to a barrier penalty in continuous optimization [28]. From this perspective, stable AMOR can be seen as a constrained optimization procedure that minimizes the cross-entropy. In that sense, if $\theta^\star$ denotes a solution to this optimization problem, the *relabeled target* $\pi_{\theta^\star} \propto \mathbb{1}_{V_{\theta^\star}} \pi$ is the restriction of π to one of its symmetric modes $V_{\theta^\star}$ that looks as Gaussian as possible.

Vector quantization algorithms have already been investigated with stochastic approximation tools [100]. However, stability was guaranteed in previous work by making strong assumptions on the trajectories of the process $(\theta_t)_{t\geq 0}$, such as in [100, Theorem 32], see also [100, Results 33 to 37 & Remark 38]. These assumptions ensure (θ_t) stays asymptotically away from sets where the function used elsewhere as a Lyapunov function is not differentiable. We adopt a different strategy by introducing the modifications of the stable AMOR algorithm and adding a barrier term in the definition of our Lyapunov function (10.1.11) that penalizes these sets. One of the contributions of this chapter is to show that this penalization strategy leads to a stable algorithm, without requiring any strong assumption on (θ_t).

10.2 Proofs

Throughout the proofs, let $\Delta_\pi > 0$ be such that

$$x \in \mathbb{X} \Rightarrow \|x\| \leq \Delta_\pi \ . \tag{10.2.1}$$

For any function $f : D \to \mathbb{R}$, we will denote by $\|f\|_\infty = \sup_{x\in D} |f(x)|$.

10.2.1 A preliminary result

First, we will use extensively the following lemma, which in particular gives the normalization constant of π_θ in (10.1.7).

Lemma 5. *Let $\theta \in \Theta$. Then*

1. *The sets $\{PV_\theta, P \in \mathcal{P}\}$ cover $\mathbb{X}$ and for any $P, Q \in \mathcal{P}$ such that $P \neq Q$, the Lebesgue measure of $PV_\theta \cap QV_\theta$ is zero.*
2. *Let λ be a measure on $(\mathbb{X}, \mathcal{X})$ with a density w.r.t. the Lebesgue measure. Furthermore, let λ be such that for any $A \in \mathcal{X}$ and $\mathcal{P} \in \mathcal{P}$, $\lambda(PA) = \lambda(A)$. Then $\lambda(V_\theta) = \lambda(\mathbb{X})/|\mathcal{P}|$.*

Proof. (1) Let $\theta \in \Theta$. We first prove that for any $P, Q \in \mathcal{P}$ and $P \neq Q$, the Lebesgue measure of $PV_\theta \cap QV_\theta$ is zero. Observe that $PV_\theta \cap QV_\theta \subseteq \{x : L_\theta(P^T x) = L_\theta(Q^T x)\}$ and $L_\theta(P^T x) = L_\theta(Q^T x)$ iff

$$(x - P\mu)^T P\Sigma^{-1}P^T(x - P\mu) = (x - Q\mu)^T Q\Sigma^{-1}Q^T(x - Q\mu) \ ,$$

or, equivalently,

$$x^T \left(P\Sigma^{-1}P^T - Q\Sigma^{-1}Q^T\right) x - 2\mu^T \left(\Sigma^{-1}P^T - \Sigma^{-1}Q^T\right) x = 0 \ .$$

Then $\{x : L_\theta(P^T x) = L_\theta(Q^T x)\}$ is either a quadratic or a linear hypersurface, and thus of Lebesgue measure zero, except if both $\Sigma^{-1} = R^T\Sigma^{-1}R$ and $\Sigma^{-1}\mu = R\Sigma^{-1}\mu$

with $R = Q^T P$. Since $\mathcal{P}$ is a group, $R \in \mathcal{P}$ and the definition (10.1.1) of Θ now guarantees that these two conditions never simultaneously hold when $\theta \in \Theta$.

We now prove that $\mathcal{X} \subseteq \bigcup_{P\in\mathcal{P}} PV_\theta$. For any $x \in \mathcal{X}$, there exists $P \in \mathcal{P}$ such that $L_\theta(Px) = \min_{Q\in\mathcal{P}} L_\theta(Qx)$. Then $x \in P^T V_\theta$ and this concludes the proof since $\mathcal{P}$ is a group.
(2) Let $\theta \in \Theta$. Using item *(1)*, it holds

$$\lambda(\mathbb{X}) = \int_{\mathbb{X}} d\lambda = \sum_{P\in\mathcal{P}} \int_{PV_\theta} d\lambda = \sum_{P\in\mathcal{P}} \int_{V_\theta} d\lambda = |\mathcal{P}| \int_{V_\theta} d\lambda \ .$$

□

10.2.2 Differentiating the cross-entropy term in (10.1.11)

Now, for $\theta \in \Theta$, let

$$\widetilde{w}(\theta) = -\int \log \mathcal{N}(x|\theta)\pi_\theta(x)\, dx \ . \tag{10.2.2}$$

Anticipating that we will need to differentiate the function w defined in (10.1.11), of which $\widetilde{w}$ is the first term, we state and prove three lemmas and a proposition that yield the gradient of $\widetilde{w}$. Lemma 6 explicitely reformulates $\widetilde{w}$ as a distortion measure in vector quantization [65]. Lemma 7 gives the gradient of a distortion measure for generic loss functions L_θ and a generic open set Θ. Its proof is adapted from [65, Lemma 4.10, page 44]. We then show in Lemma 8 that Lemma 7 applies to the loss function given by (9.2.1) and the set Θ given by (10.1.1). Finally, Proposition 9 gives an expression of the gradient of $\widetilde{w}$.

Lemma 6. *For any $\theta \in \Theta$,*

$$\widetilde{w}(\theta) = \frac{1}{2} \ln \det(\Sigma) + \frac{1}{2} \int \min_{P\in\mathcal{P}} L_{(P\mu, P\Sigma P^T)}(x)\, \pi(x)\, dx \ .$$

Proof. Let $\theta \in \Theta$. By definition of $\widetilde{w}$ and by Lemma 5,

$$\widetilde{w}(\theta) = \frac{1}{2} \ln \det(\Sigma) + \frac{|\mathcal{P}|}{2} \int_{V_\theta} L_\theta(x)\pi(x)\, dx \ ,$$

where V_θ and L_θ are given respectively by (9.2.2) and (9.2.1). Upon noting that π is invariant under the action of $\mathcal{P}$, we compute

$$|\mathcal{P}| \int_{V_\theta} L_\theta(x)\pi(x)\, dx = \sum_{P\in\mathcal{P}} \int_{V_\theta} L_\theta(x)\pi(x)\, dx = \sum_{P\in\mathcal{P}} \int_{PV_\theta} L_\theta(P^T x)\pi(x)\, dx \ .$$

In addition, by the definition (9.2.2) of V_θ,

$$PV_\theta = \{x \in \mathcal{X} : L_\theta(P^T x) = \min_{Q\in\mathcal{P}} L_\theta(Qx)\} \ .$$

Then by Lemma 5,

$$|\mathcal{P}|\int_{V_\theta} L_\theta(x)\pi(x)\,dx = \sum_{P\in\mathcal{P}}\int_{PV_\theta}\min_{Q\in\mathcal{P}} L_\theta(Qx)\pi(x)\,dx = \int \min_{Q\in\mathcal{P}} L_\theta(Qx)\pi(x)\,dx\;.$$

Finally, by the definition (9.2.1) of L_θ, $L_\theta(Qx) = L_{(Q^T\mu,Q^T\Sigma Q)}(x)$, and this concludes the proof. □

Lemma 7. *Let Θ be an open subset of $\mathbb{R}^\ell$, r be a positive integer and $\mathcal{O}\subseteq\Theta^r$ be an open set. Let $\mathcal{X}\subseteq\mathbb{R}^d$ be a measurable set and π be a probability density w.r.t. the Lebesgue measure on $\mathcal{X}$. Let $\{L_\theta,\theta\in\Theta\}$ be a family of loss functions $L_\theta:\mathcal{X}\to\mathbb{R}$, satisfying*

A. For π-almost every x, $\theta\mapsto L_\theta(x)$ is C^1 on Θ and for any $\theta\in\Theta$, there exists $h_0>0$ such that

$$\int \sup_{\|h\|\le h_0}\frac{1}{\|h\|}|h^T\,\nabla_\theta L_\theta(x)|\;\pi(x)\,dx<\infty\;.$$

B. For any $\theta\in\Theta$, there exists $h_0>0$ such that

$$\int \sup_{\|h\|\le h_0}\frac{|L_{\theta+h}(x)-L_\theta(x)|}{\|h\|}\,\pi(x)\,dx<\infty\;.$$

C. For any $\boldsymbol{\theta}=(\theta_1,...,\theta_r)\in\mathcal{O}$, the sets

$$V_{\theta_i}=\{x\in\mathcal{X}:L_{\theta_i}(x)\le \min_j L_{\theta_j}(x)\}$$

are measurable, cover $\mathcal{X}$ and for any $i\neq j$, the Lebesgue measure of $V_{\theta_i}\cap V_{\theta_j}$ is zero.

For $\boldsymbol{\theta}=(\theta_1,\cdots,\theta_r)\in\mathcal{O}$ define the function $\varphi:\Theta^r\to\mathbb{R}$ by

$$\varphi(\boldsymbol{\theta})=\int\min_{1\le i\le r}L_{\theta_i}(x)\;\pi(x)\,dx\;.$$

Then φ is differentiable on $\mathcal{O}$ and for $1\le i\le r$,

$$\nabla_{\theta_i}\varphi(\boldsymbol{\theta})=\int_{V_{\theta_i}}\nabla_{\theta_i}L_{\theta_i}(x)\,\pi(x)\,dx\;.$$

Proof. Let $\boldsymbol{\theta}=(\theta_1,\cdots,\theta_r)\in\mathcal{O}$. Set

$$d(x,\boldsymbol{\theta})=\min_{1\le i\le r}L_{\theta_i}(x)\;.$$

By definition of the function φ

$$\varphi(\boldsymbol{\theta}+\mathbf{h})-\varphi(\boldsymbol{\theta})=\int\big(d(x,\boldsymbol{\theta}+\mathbf{h})-d(x,\boldsymbol{\theta})\big)\,\pi(x)\,dx\;. \tag{10.2.3}$$

We now prove that

$$\lim_{\|h\|\to 0} \|h\|^{-1} \left(\varphi(\boldsymbol{\theta}+\mathbf{h}) - \varphi(\boldsymbol{\theta}) - \sum_{i=1}^{r} \int_{V_{\theta_i}} \langle \nabla_{\theta_i} L_{\theta_i}(x), h_i \rangle \, \pi(x)\, dx \right) = 0$$

by applying the dominated convergence theorem. First, by Assumption C,

$$\begin{aligned} \varphi(\boldsymbol{\theta}+\mathbf{h}) - \varphi(\boldsymbol{\theta}) \quad & - \quad \sum_{i=1}^{r} \int_{V_{\theta_i}} \langle \nabla_{\theta_i} L_{\theta_i}(x), h_i \rangle \, \pi(x)\, dx \\ & = \quad \sum_{i=1}^{r} \int_{V_{\theta_i}} \big(d(x, \boldsymbol{\theta}+\mathbf{h}) - d(x, \boldsymbol{\theta}) - \langle \nabla_{\theta_i} L_{\theta_i}(x), h_i \rangle \big)\, \pi(x)\, dx \; . \end{aligned}$$

Now set

$$V_{\theta_i}^{\circ} = \{ x \in \mathcal{X} : L_{\theta_i}(x) < \min_{j \neq i} L_{\theta_j}(x) \}$$

and note that $V_{\theta_i} \setminus V_{\theta_i}^{\circ}$ has measure zero under Assumption C. Then

$$\begin{aligned} \varphi(\boldsymbol{\theta}+\mathbf{h}) - \varphi(\boldsymbol{\theta}) - & \sum_{i=1}^{r} \int_{V_{\theta_i}} \langle \nabla_{\theta_i} L_{\theta_i}(x), h_i \rangle \, \pi(x)\, dx \\ & = \sum_{i=1}^{r} \int_{V_{\theta_i}^{\circ}} \big(d(x, \boldsymbol{\theta}+\mathbf{h}) - d(x, \boldsymbol{\theta}) - \langle \nabla_{\theta_i} L_{\theta_i}(x), h_i \rangle \big)\, \pi(x)\, dx \; . \end{aligned}$$

Let $x \in V_{\theta_i}^{\circ}$; under Assumption A, $\theta \mapsto L_\theta(x)$ is continuous on Θ and there exists ε_x such that

$$\|h\| \leq \varepsilon_x \Rightarrow d(x, \boldsymbol{\theta}+\mathbf{h}) = L_{\theta_i + h_i}(x) \; .$$

Then, by Assumption A,

$$\begin{aligned} d(x, \boldsymbol{\theta}+\mathbf{h}) - d(x, \boldsymbol{\theta}) - \langle \nabla_{\theta_i} L_{\theta_i}(x), h_i \rangle \quad & = \quad L_{\theta_i+h_i}(x) - L_{\theta_i}(x) - \langle \nabla_{\theta_i} L_{\theta_i}(x), h_i \rangle \\ & = \quad C(\theta_i, x, h_i) \end{aligned}$$

with $\|h_i\|^{-1} C(\theta_i, x, h_i) \to 0$ when $\|h_i\| \to 0$. Hence, we proved that for any $i \leq r$ and any $x \in V_{\theta_i}^{\circ}$,

$$\lim_{\|h\|\to 0} \|h\|^{-1} \big(d(x, \boldsymbol{\theta}+\mathbf{h}) - d(x, \boldsymbol{\theta}) - \langle \nabla_{\theta_i} L_{\theta_i}(x), h_i \rangle \big) = 0 \; .$$

We now prove that there exists h_0 such that

$$\int \sup_{\|h\| \leq h_0} \|h\|^{-1} \Big| d(x, \boldsymbol{\theta}+\mathbf{h}) - d(x, \boldsymbol{\theta}) - \sum_{i=1}^{r} \langle \nabla_{\theta_i} L_{\theta_i}(x), h_i \rangle \, \mathbb{1}_{V_{\theta_i}}(x) \Big| \, \pi(x)\, dx < +\infty \; . \tag{10.2.4}$$

First remark that for all $z, \mathbf{a} = (a_1, \cdots, a_r)$, $\mathbf{b} = (b_1, \cdots, b_r)$,

$$|d(z, \mathbf{a}+\mathbf{b}) - d(z, \mathbf{a})| \leq \max_{1 \leq i \leq r} |L_{a_i+b_i}(z) - L_{a_i}(z)| \; . \tag{10.2.5}$$

Indeed, assume without loss of generality that $d(z, \mathbf{a}) \leq d(z, \mathbf{a}+\mathbf{b})$ and let i be such that $d(z, \mathbf{a}) = L_{a_i}(z)$, then by definition of the distance d, $d(z, \mathbf{a}+\mathbf{b}) \leq L_{a_i+b_i}(z)$, which proves Eq. (10.2.5). Now, the proof of (10.2.4) is a consequence of Assumptions A and B and the inequality

$$\max_{1\leq i\leq r} |L_{a_i+b_i}(z) - L_{a_i}(z)| \leq \sum_{i=1}^{r} |L_{a_i+b_i}(z) - L_{a_i}(z)| \;.$$

□

Lemma 8. *Under Assumption 1, the quadratic loss function given by (9.2.1), the set Θ given by* (10.1.1)*, and the open set*

$$\mathcal{O} = \{(P\mu, P\Sigma P^T) : P \in \mathcal{P}, (\mu, \Sigma) \in \Theta\}$$

satisfy the assumptions of Lemma 7.

Proof. When taking derivatives with respect to a matrix, we shall use the "vec" notation during computations. For a $d \times d$ matrix A, its vectorized form $\mathrm{vec}(A)$ is a d^2 vector such that $\mathrm{vec}(A)$ stacks the columns of A on top of one another. In general, we refer to [31] for matrix algebra notions.

We check the conditions of Lemma 7. Denote by r the cardinality of $\mathcal{P}$ and set $\mathcal{P} = (I_d, P_2, \cdots, P_r)$, where I_d is the $d \times d$ identity matrix. We set

$$\mathcal{O} = \{(\theta_1, \cdots, \theta_r) \in \Theta^r : \theta_i = (P_i\mu, P_i\Sigma P_i^T), \forall i \geq 1\} \;.$$

Note that for $\boldsymbol{\theta} \in \mathcal{O}$, $L_{\theta_i}(x) = L_{\theta_1}(P_i^T x)$ and $V_{\theta_i} = P_i V_{\theta_1}$. Now, we have

$$(\mu, \Sigma) \mapsto (x-\mu)^T \Sigma^{-1}(x-\mu) = \frac{1}{\det \Sigma}(x-\mu)^T \mathrm{Adjugate}(\Sigma)(x- \;\; u)$$

so that $\theta \mapsto L_\theta(x)$ is a rational function in the coefficients of μ and Σ whose denominator $\det \Sigma > 0$. In addition,

$$\sup_{\|h\|\leq h_0} \frac{1}{\|h\|} \left|h^T \nabla_\theta L_\theta(x)\right| \leq \|\nabla_\theta L_\theta(x)\| \leq \|\nabla_\mu L_\theta(x)\| + \|\nabla_\Sigma L_\theta(x)\| \;.$$

The RHS is at most quadratic in x (for fixed θ). By Assumption 1, the RHS is π-integrable. This proves Assumption A of Lemma 7.

We now prove Assumption B of Lemma 7. Let $\theta \in \Theta$ and set $\Delta\theta = (\Delta\mu, \Delta\Sigma)$. By standard algebra, we have

$$(\Sigma + \Delta\Sigma)^{-1} = \Sigma^{-1} - \Sigma^{-1}\,\Delta\Sigma\,\Sigma^{-1} + o(\|\Delta\Sigma\|)$$

for any matrix $\Delta\Sigma$ such that $\Sigma + \Delta\Sigma$ is invertible. Therefore,

$$L_{\theta+\Delta\theta}(x) - L_\theta(x) = -2(\Delta\mu)^T\Sigma^{-1}(x-\mu) - (x-\mu)^T\Sigma^{-1}\,\Delta\Sigma\,\Sigma^{-1}(x-\mu) + \Xi(x, \theta, \Delta\theta) \;,$$

for some function $\Xi(x, \theta, \Delta\theta)$ such that

$$|\Xi(x, \theta, \Delta\theta)| \leq C(\theta)\|x\|^2\|\Delta\theta\|^2$$

and some constant $C(\theta)$ (depending upon θ but independent of x and $\Delta\theta$). The proof is concluded since, by Assumption 1, $\int \|x\|^2\pi(x)\, dx < +\infty$.

Finally, the sets V_{θ_i} are measurable for any $\theta_1, \cdots, \theta_r \in \Theta$ since $(x, \theta) \mapsto L_\theta(x)$ is continuous on $\mathcal{X} \times \Theta$. The proof of Assumption C of Lemma 7 is then concluded by application of Lemma 5. □

We are now ready to state the final result of this preliminary section, and give the expression of the gradient of $\widetilde{w}$ defined in (10.2.2).

Proposition 9. *Under Assumption 1, the function $\widetilde{w}$ defined in (10.2.2) is continuously differentiable on Θ and for any $\theta \in \Theta$,*

$$\begin{aligned}\nabla_\mu \widetilde{w}(\theta) &= -\Sigma^{-1}(\mu_{\pi_\theta} - \mu)\ ,\\ \nabla_\Sigma \widetilde{w}(\theta) &= -\frac{1}{2}\Sigma^{-1}\left(\Sigma_{\pi_\theta} - \Sigma + (\mu_{\pi_\theta} - \mu)(\mu_{\pi_\theta} - \mu)^T\right)\Sigma^{-1}\ .\end{aligned}$$

Proof. Let r denote the cardinality of $\mathcal{P}$ and set $\mathcal{P} = (I_d, P_2, \cdots, P_r)$. Let $\theta \in \Theta$. By Lemma 6, we have

$$\widetilde{w}(\theta) = \frac{1}{2}\ln\det(\Sigma) + \frac{1}{2}\int \min_{1\leq i\leq r} L_{\theta_i}(x)\ \pi(x)\, dx\ ,$$

where $\theta_i = (P_i\mu, P_i\Sigma^{-1}P_i^T)$.

We first consider the derivative w.r.t. μ. We have

$$\nabla_\mu \widetilde{w}(\theta) = \frac{1}{2}\nabla_\mu \int \min_{1\leq i\leq r} L_{\theta_i}(x)\ \pi(x)\, dx.$$

By Lemmas 7 and 8 and the chain rule, we have

$$\begin{aligned}\nabla_\mu \widetilde{w}(\theta) &= \frac{1}{2}\sum_{i=1}^r P_i^T \int_{A_i} \nabla_{\mu_i}\left[(x - \mu_i)P_i\Sigma^{-1}P_i^T(x - \mu_i)\right]_{\mu_i = P_i\mu}\ \pi(x)\, dx\\ &= -\Sigma^{-1}\sum_{i=1}^r \int_{A_i} (P_i^T x - \mu)\ \pi(x)\, dx\ ,\end{aligned}$$

where

$$A_i = \{x : L_{\theta_i}(x) \leq \min_j L_{\theta_j}(x)\} = P_i V_\theta\ ,$$

with V_θ defined in (9.2.2). Hence, by Lemma 5, and since π is invariant under the action of $\mathcal{P}$, we have

$$\begin{aligned}\nabla_\mu \widetilde{w}(\theta) &= -\Sigma^{-1}\sum_{i=1}^r \int_{V_\theta} (x - \mu)\ \pi(x)\, dx\\ &= -\Sigma^{-1}\int (x - \mu)[r\,\pi(x)\,\mathbb{1}_{V_\theta}(x)]\, dx\\ &= -\Sigma^{-1}\left(\mu_{\pi_\theta} - \mu\right)\ ,\end{aligned}$$

where we used the definition (10.1.8) of μ_{π_θ}.

We now consider the derivative w.r.t. Σ, that we will derive in a similar manner. We refer to [31] for matrix algebra notions such as Kronecker products. First remark that, by standard algebra and since Σ is symmetric,

$$\nabla_{\mathrm{vec}(\Sigma)} \ln \det \Sigma = \mathrm{vec}(\Sigma^{-1}) \ .$$

Then recall that

$$\nabla_{\mathrm{vec}(\Sigma)}(x-\mu)\Sigma^{-1}(x-\mu) = -\Sigma^{-1}(x-\mu)\otimes\Sigma^{-1}(x-\mu) \ .$$

Now let, for A a matrix, $A^{\otimes 2} = A \otimes A$. Using Lemmas 7 and 8 along with the chain rule, we compute

$$\begin{aligned}
&\nabla_{\mathrm{vec}(\Sigma)}\widetilde{w}(\theta) - \frac{1}{2}\mathrm{vec}(\Sigma^{-1}) \\
&\quad = \frac{1}{2}\sum_{i=1}^{r}(P_i^{\otimes 2})^T \int_{P_i V_\theta} \nabla_{\mathrm{vec}(\Sigma_i)} \left[(x-P_i\mu)^T\Sigma_i^{-1}(x-P_i\mu)\right]_{\Sigma_i = P_i\Sigma P_i^T} \pi(x)\,dx \\
&\quad = -\frac{1}{2}\sum_{i=1}^{r}(P_i^T)^{\otimes 2}\int_{P_i V_\theta}\left[P_i\Sigma^{-1}P_i^T(x-P_i\mu)\right]^{\otimes 2}\pi(x)\,dx \\
&\quad = -\frac{1}{2}\sum_{i=1}^{r}\int_{P_i V_\theta}\left[\Sigma^{-1}(P_i^T x-\mu)\right]^{\otimes 2}\pi(x)\,dx \\
&\quad = -\frac{1}{2}(\Sigma^{-1})^{\otimes 2}\sum_{i=1}^{r}\int_{P_i V_\theta}[P_i^T x-\mu]^{\otimes 2}\pi(x)\,dx
\end{aligned}$$

where we used the identities $(A\otimes B)^T = A^T\otimes B^T$ and $(A\otimes B)(C\otimes D) = (AC)\otimes(BD)$. A change of variables now leads to

$$\begin{aligned}
&\nabla_{\mathrm{vec}(\Sigma)}\widetilde{w}(\theta) - \frac{1}{2}\mathrm{vec}(\Sigma^{-1}) \\
&\quad = -\frac{1}{2}(\Sigma^{-1})^{\otimes 2}\sum_{i=1}^{r}\int_{V_\theta}(x-\mu)^{\otimes 2}\pi(x)\,dx \\
&\quad = -\frac{1}{2}(\Sigma^{-1})^{\otimes 2}\int(x-\mu_{\pi_\theta}+\mu_{\pi_\theta}-\mu)^{\otimes 2}[r\,\pi(x)\,\mathbb{1}_{V_\theta}(x)]\,dx \\
&\quad = -\frac{1}{2}(\Sigma^{-1})^{\otimes 2}\left(\int(x-\mu_{\pi_\theta})\otimes(x-\mu_{\pi_\theta})\pi_\theta(x)\,dx + (\mu_{\pi_\theta}-\mu)\otimes(\mu_{\pi_\theta}-\mu)\right) \\
&\quad = -\frac{1}{2}(\Sigma^{-1}\otimes\Sigma^{-1})\mathrm{vec}(\Sigma_{\pi_\theta}+(\mu_{\pi_\theta}-\mu)(\mu_{\pi_\theta}-\mu)^T) \ ,
\end{aligned}$$

where we used the distributivity of the Kronecker product, Lemma 5, and the definitions (10.1.8) and (10.1.9) of μ_{π_θ} and Σ_{π_θ}. Finally, the identity $\mathrm{vec}(AXB) = (B^T\otimes A)\mathrm{vec}(X)$ allows us to write

$$\nabla_{\mathrm{vec}(\Sigma)}\widetilde{w}(\theta) = -\frac{1}{2}\mathrm{vec}\big(\Sigma^{-1}\left[\Sigma_{\pi_\theta}-\Sigma+(\mu_{\pi_\theta}-\mu)(\mu_{\pi_\theta}-\mu)^T\right]\Sigma^{-1}\big) \ .$$

□

10.2.3 The Lyapunov function

Lemma 10 establishes the existence of a *Lyapunov* function for the mean field h given by (10.1.10).

Lemma 10. *Under Assumption 1, the mean field h is continuous on Θ, the function w defined by (10.1.11) is $\mathcal{C}^1$ on Θ and*

1. $\nabla_\mu w(\theta) = -\Sigma^{-1} h_\mu(\theta)$ *and* $\nabla_\Sigma w(\theta) = -\frac{1}{2}\Sigma^{-1} h_\Sigma(\theta)\Sigma^{-1}$.

2. $\langle \nabla w(\theta), h(\theta)\rangle \leq 0$ *on Θ and* $\langle \nabla w(\theta), h(\theta)\rangle = 0$ *iff $\theta \in \mathcal{L}$.*

3. *For any $M > 0$, the level set*

$$\mathcal{W}_M = \{\theta \in \Theta : w(\theta) \leq M\} \tag{10.2.6}$$

is a compact subset of Θ, and there exist $\delta_1, \delta_2 > 0$ such that

$$\inf_{\theta \in \mathcal{W}_M} \inf_{P \in \mathcal{P}^*} \|(I-P)\Sigma^{-1}\mu\| \geq \delta_1 \textit{ and} \tag{10.2.7a}$$

$$\inf_{\theta \in \mathcal{W}_M} \lambda_{\min}(\Sigma) \geq \delta_2 \ , \tag{10.2.7b}$$

where $\lambda_{\min}(\Sigma)$ denotes the minimal eigenvalue of the real symmetric matrix Σ.

Remark 11. As a consequence of Lemma 10, observe that for any $M > 0$, there exists $\delta > 0$ such that $\mathcal{W}_M \subseteq \mathcal{K}_\delta$, where $\mathcal{K}_\delta$ is defined in (10.1.3).

Proof. (Continuity of h) Since $(I-P)\Sigma^{-1}\mu \neq 0$ on Θ for any $P \in \mathcal{P}^*$, it suffices to show that $\theta \mapsto \mu_{\pi_\theta}$ and $\theta \mapsto \Sigma_{\pi_\theta}$ are continuous. Since, by Lemma 5, the boundary of V_θ is of Lebesgue measure zero, the continuity of $\theta \mapsto \mu_{\pi_\theta}$ follows from Lebesgue's dominated convergence theorem if, for any $x \in \mathbb{X} \setminus \partial V_\theta$, $\theta \mapsto x\mathbb{1}_{V_\theta}(x)$ is continuous. To see this, note that if x is in the interior of V_θ, then there exists a neighborhood $\mathcal{V}$ of θ such that for any $\theta' \in \mathcal{V}$, $x \in V_{\theta'}$, and if $x \in \mathbb{X} \setminus V_\theta$, which is an open subset of $\mathbb{X}$, then there exists a neighborhood $\mathcal{V}$ of θ such that for any $\theta' \in \mathcal{V}$, $x \in \mathbb{X} \setminus V_{\theta'}$.

The case of $\theta \mapsto \Sigma_\theta$ is similar and omitted.

(w is C^1 on Θ) Proposition 9 states that the first term in the RHS of (10.1.11) is continuously differentiable on Θ. Since $\|(I-P)\Sigma^{-1}\mu\| \neq 0$ for any $P \in \mathcal{P}^*$ and $(\mu, \Sigma) \in \Theta$, the second term in the RHS of (10.1.11) is also continuously differentiable on Θ. By Proposition 9, it thus holds, for any $\theta = (\mu, \Sigma) \in \Theta$,

$$\begin{aligned}\nabla_\mu w(\theta) &= -\Sigma^{-1}(\mu_{\pi_\theta} - \mu) + \alpha \sum_P \frac{1}{\|(I-P)\Sigma^{-1}\mu\|^4}\, \Sigma^{-1} U_P \Sigma^{-1}\mu \\ &= -\Sigma^{-1} h_\mu(\theta)\end{aligned}$$

and

$$\begin{aligned}\nabla_\Sigma w(\theta) &= -\frac{1}{2}\Sigma^{-1}(\Sigma_{\pi_\theta} - \Sigma + (\mu-\mu_{\pi_\theta})(\mu-\mu_{\pi_\theta})^T)\Sigma^{-1}\dots \\ &\quad -\frac{\alpha}{2}\sum_P \frac{1}{\|(I-P)\Sigma^{-1}\mu\|^4}\;\Sigma^{-1}\left(\mu\mu^T\Sigma^{-1}U_P\right)\Sigma^{-1} + U_P\Sigma^{-1}\mu\mu^T \\ &= -\frac{1}{2}\Sigma^{-1}h_\Sigma(\theta)\Sigma^{-1}\;.\end{aligned}$$

Hence, upon noting that $h_\Sigma(\theta)$ and Σ^{-1} are symmetric,

$$\begin{aligned}\langle\nabla w(\theta), h(\theta)\rangle &= -h_\mu(\theta)^T\Sigma^{-1}h_\mu(\theta) - \frac{1}{2}\mathrm{Trace}\Big(\Sigma^{-1}h_\Sigma(\theta)\Sigma^{-1}h_\Sigma(\theta)\Big) \\ &= -h_\mu(\theta)^T\Sigma^{-1}h_\mu(\theta) - \frac{1}{2}\mathrm{Trace}\Big(\Sigma^{-1/2}h_\Sigma(\theta)\Sigma^{-1}h_\Sigma(\theta)\Sigma^{-1/2}\Big).\end{aligned}$$

The first term of the RHS is negative since $\Sigma \in \mathcal{C}_d^+$ and the second term is negative since $(A,B) \mapsto \mathrm{Trace}(A^TB)$ is a scalar product. Therefore $\langle\nabla w(\theta), h(\theta)\rangle \le 0$ with equality iff $\theta \in \mathcal{L}$.

($\mathcal{W}_M$ is compact) We prove (10.2.7a). By the definition (10.1.11) of w, for any $\theta \in \mathcal{W}_M$, we have

$$-\int \log\mathcal{N}(x|\theta)\pi_\theta(x)\,dx + \frac{\alpha}{2}\sum_{P\in\mathcal{P}^*}\frac{1}{\|(I-P)\Sigma^{-1}\mu\|^2} \le M\;.$$

In particular, since the first term in the LHS is a cross-entropy and thus it is non-negative (alternatively, see [14, Proposition 1 of the supplementary material]), for any $\theta \in \mathcal{W}_M$, we have

$$\sum_{P\in\mathcal{P}^*}\frac{1}{\|(I-P)\Sigma^{-1}\mu\|^2} \le \frac{2M}{\alpha}\;.$$

This yields $\|(I-P)\Sigma^{-1}\mu\|^2 \ge \frac{\alpha}{2M}$ for any $P \in \mathcal{P}^*$, thus concluding the proof of (10.2.7a).

We now prove (10.2.7b). Let $\theta = (\mu, \Sigma) \in \mathcal{W}_M$. Denote by $(\lambda_i(\Sigma))_{i\le d}$ the eigenvalues of Σ. Since Σ is symmetric, there exist $d\times d$ matrices Q_θ, Λ_θ such that $\Sigma = Q_\theta\Lambda_\theta Q_\theta^T$, Q_θ is orthogonal, and $\Lambda_\theta = \mathrm{Diag}(\lambda_i(\Sigma))$. Then

$$\begin{aligned}2M &\ge 2w(\theta) \ge -2\int\log\mathcal{N}(x|\theta)\pi_\theta(x)\;dx \\ &= d\log(2\pi) + \log\det\Sigma + (\mu_{\pi_\theta}-\mu)^T\Sigma^{-1}(\mu_{\pi_\theta}-\mu) + \mathrm{Trace}(\Sigma^{-1}\Sigma_{\pi_\theta}) \qquad (10.2.8) \\ &\ge \sum_{i=1}^d \log\lambda_i(\theta) + 0 + \mathrm{Trace}(\Sigma^{-1}\Sigma_{\pi_\theta})\;.\end{aligned}$$

Set $b_i(\theta) = (Q_\theta^T\Sigma_{\pi_\theta}Q_\theta)_{ii}$. Then

$$\mathrm{Trace}(\Sigma^{-1}\Sigma_{\pi_\theta}) = \mathrm{Trace}(Q_\theta\Lambda_\theta^{-1}Q_\theta^T\Sigma_{\pi_\theta}) = \mathrm{Trace}(Q_\theta^T\Sigma_{\pi_\theta}Q_\theta\Lambda_\theta^{-1}) = \sum_{i=1}^d\frac{b_i(\theta)}{\lambda_i(\theta)}\;. \tag{10.2.9}$$

Therefore, for any $\theta \in \mathcal{W}_M$,

$$\sum_{i=1}^{d} \log \lambda_i(\theta) + \frac{b_i(\theta)}{\lambda_i(\theta)} \leq 2M \ . \tag{10.2.10}$$

We now prove that for any i, $\inf_{\mathcal{W}_M} b_i > 0$. This property, combined with (10.2.10), will conclude the proof of (10.2.7b). Let $\varepsilon > 0$ be such that

$$2^d \varepsilon \|\pi\|_\infty \Delta_\pi^{d-1} < |\mathcal{P}| \ ,$$

and for $v \in \{x \in \mathbb{R}^d : \|x\| = 1\}$, let

$$B_\varepsilon^v(\theta) = \{x \in \mathrm{Supp}(\pi) \cap V_\theta : |\langle x - \mu_{\pi_\theta}, v\rangle| \leq \varepsilon\} \ . \tag{10.2.11}$$

Note that by Assumption 1,

$$\pi\big(B_\varepsilon^v(\theta)\big) \leq \|\pi\|_\infty \mathrm{Leb}\big(B_\varepsilon^v(\theta)\big) \leq 2^d \varepsilon \|\pi\|_\infty \Delta_\pi^{d-1} \ .$$

Then, by definition of ε,

$$\pi\big(V_\theta \setminus B_\varepsilon^v(\theta)\big) \geq |\mathcal{P}| - 2^d \varepsilon \|\pi\|_\infty \Delta_\pi^{d-1} > 0 \ . \tag{10.2.12}$$

Now, if (e_i) denotes the canonical basis of $\mathbb{R}^d$, then

$$\begin{aligned} b_i(\theta) &= |\mathcal{P}| e_i^T Q_\theta^T \left(\int_{V_\theta} (x - \mu_{\pi_\theta})(x - \mu_{\pi_\theta})^T \pi(x) \ dx \right) Q_\theta e_i \\ &= |\mathcal{P}| \int_{V_\theta} (Q_\theta e_i)^T (x - \mu_{\pi_\theta})(x - \mu_{\pi_\theta})^T Q_\theta e_i \ \pi(x) \ dx \\ &= |\mathcal{P}| \int_{V_\theta} \langle x - \mu_{\pi_\theta}, Q_\theta e_i\rangle^2 \pi(x) \ dx \\ &\geq |\mathcal{P}| \int_{V_\theta \setminus B_\varepsilon^{Q_\theta e_i}(\theta)} \langle x - \mu_{\pi_\theta}, Q_\theta e_i\rangle^2 \pi(x) \ dx \\ &\geq \varepsilon^2 |\mathcal{P}| \pi\big(V_\theta \setminus B_\varepsilon^{Q_\theta e_i}(\theta)\big) \ , \end{aligned} \tag{10.2.13}$$

where the last inequality follows from the definition (10.2.11) of $B_\varepsilon^{Q_\theta e_i}(\theta)$. Thus, by (10.2.12), $b_i(\theta)$ is bounded away from zero on $\mathcal{W}_M$.

As w is continuous on Θ, $\{\theta \in \Theta, \ w(\theta) \leq M\}$ is closed. From (10.2.7b), (10.2.8) and Assumption 1, $\mu \mapsto (\mu_{\pi_\theta} - \mu)^T \Sigma^{-1} (\mu_{\pi_\theta} - \mu)$ is bounded on $\mathcal{W}_M$. In addition, (10.2.8), (10.2.9) and (10.2.13) imply that $\Sigma \mapsto \log\det\Sigma$ is bounded on $\mathcal{W}_M$. These properties combined with (10.2.7b) imply that $\mathcal{W}_M$ is bounded. Hence $\mathcal{W}_M$ is compact.

□

10.2.4 Proof of Proposition 1

(1) By the definition (10.1.1) of Θ and Lemma 5, $\forall \theta \in \Theta, x \in \mathbb{X}$, it holds

$$\int_{V_\theta} q_\theta(x,y) \; dy = \sum_{P \in \mathcal{P}} \int_{V_\theta} \mathcal{N}(Py|x, c\Sigma) \; dy = 1 \; .$$

(2) Let $(X_t)_{t \geq 0}$ and $(\theta_t)_{t \geq 0}$ be the random processes defined by the stable AMOR algorithm in Figure 10.1. We prove that for any measurable positive function f,

$$\mathbb{E}[f(X_t)|X_0, \theta_0, \ldots, X_{t-1}, \theta_{t-1}] = \int f(x_t) P_{\theta_{t-1}}(X_{t-1}, x_t) \; dx_t \; , w.p.1.$$

Let f be measurable and positive. Let $(\tilde{P}, \tilde{X})$ be the r.v. defined by Steps 5 and 6. Let U be a uniform r.v. independent of $\sigma(X_0, \theta_0, \ldots, X_{t-1}, \theta_{t-1}, \tilde{P}, \tilde{X})$. By construction, it holds that

$$\begin{aligned}\mathbb{E}[f(X_t)|X_0, \theta_0, \ldots, X_{t-1}, \theta_{t-1}] &= \mathbb{E}[f(\tilde{P}\tilde{X}) \mathbb{1}_{U \leq \alpha_{\theta_{t-1}}(X_{t-1}, \tilde{P}\tilde{X})} | X_0, \theta_0, \ldots, X_{t-1}, \theta_{t-1}] \\ &+ \mathbb{E}[f(X_{t-1}) \mathbb{1}_{U > \alpha_{\theta_{t-1}}(X_{t-1}, \tilde{P}\tilde{X})} | X_0, \theta_0, \ldots, X_{t-1}, \theta_{t-1}] \; ,\end{aligned} \tag{10.2.14}$$

where $\alpha_\theta(x,y)$ is given by (10.1.5). Since U is independent of the past and from $\tilde{P}$ and $\tilde{X}$, we have

$$\begin{aligned}&\mathbb{E}[f(\tilde{P}\tilde{X}) \mathbb{1}_{U \leq \alpha_{\theta_{t-1}}(X_{t-1}, \tilde{P}\tilde{X})} | X_0, \theta_0, \ldots, X_{t-1}, \theta_{t-1}] \\ &\quad = \mathbb{E}\left[f(\tilde{P}\tilde{X}) \left(1 - \alpha_{\theta_{t-1}}(X_{t-1}, \tilde{P}\tilde{X})\right) | X_0, \theta_0, \ldots, X_{t-1}, \theta_{t-1} \right] \; ,\end{aligned} \tag{10.2.15}$$

and

$$\begin{aligned}&\mathbb{E}[f(X_{t-1}) \mathbb{1}_{U > \alpha_{\theta_{t-1}}(X_{t-1}, \tilde{P}\tilde{X})} | X_0, \theta_0, \ldots, X_{t-1}, \theta_{t-1}] \\ &\quad = f(X_{t-1}) \; \mathbb{E}\left[\left(1 - \alpha_{\theta_{t-1}}(X_{t-1}, \tilde{P}\tilde{X})\right) | X_0, \theta_0, \ldots, X_{t-1}, \theta_{t-1} \right] \; .\end{aligned} \tag{10.2.16}$$

Now note that the projection mechanism (Steps 15 to 17 in Figure 10.1) guarantees that $\theta_{t-1} \in \Theta$ with probability 1. By Lemma 5, $\theta \in \Theta$ implies $\mathbb{X} = \cup_P (PV_\theta)$ and

$$\forall P, Q \in \mathcal{P} \text{ such that } P \neq Q, \; \text{Leb}(PV_\theta \cap QV_\theta) = 0.$$

Thus, for any measurable and bounded function $\varphi : \mathbb{X} \times \Theta \to \mathbb{R}$, we have

$$\int_{\mathbb{X}} \varphi(x, \theta) \, dx = \sum_{Q \in \mathcal{P}} \int_{QV_\theta \cap (\cup_{R \neq Q} RV_\theta)^c} \varphi(x, \theta) \, dx \; .$$

Applying this decomposition to (10.2.15) yields

$$\begin{aligned}&\mathbb{E}[f(\tilde{P}\tilde{X}) \mathbb{1}_{U \leq \alpha_{\theta_{t-1}}(X_{t-1}, \tilde{P}\tilde{X})} | X_0, \theta_0, \ldots, X_{t-1}, \theta_{t-1}] \\ &\quad = \sum_{P \in \mathcal{P}} \int h(Px) \frac{1}{N(x, \theta_{t-1})} \mathbb{1}_{V_{\theta_{t-1}}}(Px) \mathcal{N}(x| > X_{t-1}, c\Sigma_{t-1}) \; dx \\ &\quad = \sum_{P, Q \in \mathcal{P}} \int_{QV_{\theta_{t-1}} \cap (\cup_{R \neq Q} RV_{\theta_{t-1}})^c} h(Px) \frac{1}{N(x, \theta_{t-1})} \mathbb{1}_{V_{\theta_{t-1}}}(Px) \mathcal{N}(x|X_{t-1}, c\Sigma_{t-1}) \; dx \; ,\end{aligned}$$

where $N(x,\theta) = |\{Q \in \mathcal{P} / Qx \in V_\theta\}|$. Using Lemma 5 again,

$$\theta \in \Theta, x \notin \cup_{P \neq Q}(PV_\theta \cap QV_\theta) \Rightarrow N(x,\theta) = 1 \ ,$$

and thus

$$\begin{aligned}
&\mathbb{E}[f(\tilde{P}\tilde{X})\mathbb{1}_{U \leq \alpha_{\theta_{t-1}}(X_{t-1},\tilde{P}\tilde{X})}|X_0,\theta_0,\ldots,X_{t-1},\theta_{t-1}] \\
&\quad = \sum_{P,Q \in \mathcal{P}} \int_{QV_{\theta_{t-1}} \cap (\cup_{R \neq Q} RV_{\theta_{t-1}})^c} h(Px)\mathbb{1}_{V_{\theta_{t-1}}}(Px)\mathcal{N}(x|X_{t-1}, c\Sigma_{t-1}) \ dx \\
&\quad = \sum_{P \in \mathcal{P}} \int h(Px)\mathbb{1}_{V_{\theta_{t-1}}}(Px)\mathcal{N}(x|X_{t-1}, c\Sigma_{t-1}) \ dx \\
&\quad = \sum_{P \in \mathcal{P}} \int h(y)\mathbb{1}_{V_{\theta_{t-1}}}(y)\mathcal{N}(P^{-1}y|X_{t-1}, c\Sigma_{t-1}) \ dy \\
&\quad = \int_{V_{\theta_{t-1}}} h(y) q_{\theta_{t-1}}(X_{t-1}, y) \ dy \ ,
\end{aligned}$$

where in the last step we used the fact that $\mathcal{P}$ is a group. Similarly,

$$\begin{aligned}
&\mathbb{E}\left[\left(1 - \alpha_{\theta_{t-1}}(X_{t-1},\tilde{P}\tilde{X})\right)|X_0,\theta_0,\ldots,X_{t-1},\theta_{t-1}\right] \\
&\qquad\qquad = \int_{V_{\theta_{t-1}}} \left((1 - \alpha_{\theta_{t-1}}(X_{t-1},y)\right) \ q_{\theta_{t-1}}(X_{t-1},y) \ dy \ ;
\end{aligned}$$

and this concludes the proof.

(3) Let $\theta \in \Theta$. Eqn. (10.1.4) implies that if $x \in V_\theta$, then $P(x, V_\theta) = 1$. To prove that $\pi_\theta P_\theta = \pi_\theta$, it is sufficient to check the detailed balance condition, which states that

$$\forall A, B \subset \mathbb{X} \text{ measurable}, \int_A \pi_\theta(x)P_\theta(x,B)\,dx = \int_B \pi_\theta(y)P_\theta(y,A)\,dy \ .$$

We consider the two summands in the definition (10.1.4) separately. First, it holds that

$$\begin{aligned}
\pi_\theta(x)\alpha_\theta(x,y)q_\theta(x,y)\mathbb{1}_{V_\theta}(y) &= |P|\big(\pi(x)q_\theta(x,y) \wedge \pi(y)q_\theta(y,x)\big)\mathbb{1}_{V_\theta}(x)\mathbb{1}_{V_\theta}(y) \\
&= \pi_\theta(y)\alpha_\theta(y,x)q_\theta(y,x)\mathbb{1}_{V_\theta}(x) \ ,
\end{aligned}$$

so

$$\int_A \pi_\theta(x)\left(\int_{B \cap V_\theta} \alpha_\theta(x,y)q_\theta(x,y)\,dy\right) \ dx = \int_B \pi_\theta(y)\left(\int_{A \cap V_\theta} \alpha_\theta(y,x)q_\theta(y,x)\,dx\right) \ dy \ .$$

Secondly,

$$\begin{aligned}
&\int_A \pi_\theta(x)\mathbb{1}_B(x)\int_{V_\theta} (1 - \alpha_\theta(x,z))\,q_\theta(x,z)\,dz \ \ dx \\
&\qquad = \int_{A \cap B} \pi_\theta(x)\int_{V_\theta} (1 - \alpha_\theta(x,z))\,q_\theta(x,z)\,dz \ \ dx \\
&\qquad\qquad = \int_B \pi_\theta(y)\mathbb{1}_A(y)\int_{V_\theta} (1 - \alpha_\theta(y,z))\,q_\theta(y,z)\,dz \ .
\end{aligned}$$

This concludes the proof of the detailed balance condition.

10.2.5 Regularity in θ of the Poisson solution

Lemma 12. *1. For any $M > 0$, there exists $\rho \in (0,1)$ such that for any $x \in \mathbb{X}$ and any $\theta \in \mathcal{W}_M$, $\|P_\theta^n(x,\cdot) - \pi_\theta\|_{\mathrm{TV}} \leq 2(1-\rho)^n$.*

2. Under Assumption 1, for any $\theta \in \Theta$, there exists a solution $\hat{H}_\theta$ of the Poisson equation $g - P_\theta g = H(\cdot,\theta) - \pi_\theta H(\cdot,\theta)$. Furthermore, for any $M > 0$,

$$\sup_{\theta\in\mathcal{W}_M} \sup_{x\in\mathbb{X}} |\hat{H}_\theta(x)| < \infty \,. \tag{10.2.17}$$

Proof. (of Item 1) It is sufficient to prove that there exists $\rho \in (0,1)$ such that for any $x \in \mathbb{X}$ and $\theta \in \mathcal{W}_M$, $P_\theta(x,\cdot) \geq \rho\pi_\theta$ (see e.g. [96, Theorem 16.2.4]). By (10.1.4), for any $x \in \mathbb{X}$ and $A \in \mathcal{X}$, $P_\theta(x,A) \geq \int_{A\cap V_\theta} \alpha_\theta(x,y) q_\theta(x,y)\, dy$. By Lemma 10, there exists $a > 0$ such that for any $(\mu,\Sigma) \in \mathcal{W}_M$, any $m, z \in \mathbb{X}$, and any $P \in \mathcal{P}$, we have $\mathcal{N}(Pz|m,\Sigma) \geq a$. Thus, for any $\theta \in \mathcal{W}_M$ and $y \in V_\theta$, it holds that

$$\alpha_\theta(x,y) q_\theta(x,y) \mathbb{1}_{V_\theta}(y) \geq a|\mathcal{P}| \left(1 \wedge \frac{\pi(y)}{\pi(x)}\right) \mathbb{1}_{V_\theta}(y) \geq \frac{a}{\|\pi\|_\infty} \pi_\theta(y) \,. \tag{10.2.18}$$

Thus, we have $P_\theta(x,\cdot) \geq \rho\pi_\theta$ for any $x \in \mathbb{X}$ and $\theta \in \mathcal{W}_M$ with $\rho = a/\|\pi\|_\infty$.

(Proof of Item 2)

$$\begin{aligned} \left|\sum_n P_\theta^n \big(H(x,\theta) - \pi_\theta(H(\cdot,\theta))\big)\right| &\leq \sup_{\theta\in\mathcal{W}_M} \|H(\cdot,\theta)\|_\infty \sum_n \|P_\theta^n(x,\cdot) - \pi_\theta\|_{\mathrm{TV}} \\ &\leq 2 \sup_{\theta\in\mathcal{W}_M} \|H(\cdot,\theta)\|_\infty \rho^{-1}. \end{aligned} \tag{10.2.19}$$

Since the sup is finite by Lemma 10, the series $\sum P_\theta^n \big(H(x,\theta) - \pi_\theta(H(\cdot,\theta))\big)$ converges. Finally, note that

$$\hat{H}_\theta(x) = \sum_n P_\theta^n \big(H(x,\theta) - \pi_\theta(H(\cdot,\theta))\big)$$

is a solution of the Poisson equation, and that $\sup_{\theta\in\mathcal{W}_M, x\in\mathbb{X}} |\hat{H}_\theta(x)| < \infty$. □

Lemma 13. *Let $M > 0$ and $\kappa \in (0,1/2)$. Under Assumption 1, there exists $C > 0$ such that for any $\theta \in \mathcal{W}_M$ and $\theta' \in \Theta$, it holds that*

$$Leb(V_\theta \setminus V_{\theta'}) \leq C\|\theta - \theta'\|^{1-2\kappa} \,, \tag{10.2.20}$$

where $Leb(A)$ denotes the Lebesgue measure of the set A.

Proof. We prove that there exist $\bar{C}, \bar{h} > 0$, such that for any $\theta \in \mathcal{W}_M$ and any $\theta' \in \Theta$ such that $\|\theta - \theta'\| \leq \bar{h}$, $\mathrm{Leb}(V_\theta \setminus V_{\theta'}) \leq \bar{C}\|\theta - \theta'\|^{1-2\kappa}$. Note that since $V_\theta \subset \mathbb{X}$ and since $\mathbb{X}$ is bounded, there exists $\check{C} > 0$ such that $\mathrm{Leb}(V_\theta \setminus V_{\theta'}) \leq \check{C}$. Therefore, (10.2.20) holds with $C = \bar{C} \vee \check{C}/\bar{h}^{1-2\kappa}$.

By Lemma 10, w is uniformly continuous on $\mathcal{W}_{M+1}$, and there exists $h_0 > 0$ small enough for which

$$\big[\theta \in \mathcal{W}_M,\ \theta' \in \Theta,\ \|\theta - \theta'\| < h_0\big] \Rightarrow \forall u \in [0,1],\ \theta + u(\theta' - \theta) \in \mathcal{W}_{M+1}\ . \tag{10.2.21}$$

Let $\bar{h} \leq h_0$. Let $\theta = (\mu, \Sigma) \in \mathcal{W}_M$ and $\theta' \neq \theta$ such that $\|\theta - \theta'\| \leq \bar{h}$.

By definition of the set V_ϑ, for any $x \in V_\theta \setminus V_{\theta'}$, there exists $P \in \mathcal{P}^*$ such that $L_{\theta'}(x) - L_{\theta'}(P^T x) > 0$ and $L_\theta(x) - L_\theta(P^T x) \leq 0$. Since $\vartheta \mapsto L_\vartheta(x) - L_\vartheta(P^T x)$ is continuous on $\mathcal{W}_{M+1}$, there exists $u \in [0,1]$ depending on x, θ, θ', and P such that $L_{\theta+u(\theta'-\theta)}(x) - L_{\theta+u(\theta'-\theta)}(P^T x) = 0$. Therefore

$$V_\theta \setminus V_{\theta'} \subset \bigcup_{P \in \mathcal{P}^*} \mathcal{V}_P,$$

where

$$\mathcal{V}_P = \bigcup_{u \in [0,1]} \mathcal{Z}\big(L_{\theta+u(\theta'-\theta)}(\cdot) - L_{\theta+u(\theta'-\theta)}(P^T \cdot)\big) \cap \mathbb{X}\ ; \tag{10.2.22}$$

and $\mathcal{Z}(f)$ denotes the zeros of the function f. The proof proceeds by showing that for any $P \in \mathcal{P}^*$, $\mathcal{V}_P$ is included in a measurable set with measure $O\left(\|\theta - \theta'\|^{1-2\kappa}\right)$.

Let $P \in \mathcal{P}^*$. Let $B(0, \Delta_\pi) = \{y \in \mathbb{R}^d : \|y\| \leq \Delta_\pi\}$, where Δ_π is defined by 10.2.1. For any $x \in B(0, \Delta_\pi)$, define

$$\begin{aligned} l_\theta(x) &= 2\mu^T \Sigma^{-1}(I - P^T)x\ , \\ q_\theta(x) &= x^T(\Sigma^{-1} - P\Sigma^{-1}P^T)x\ , \\ \mathsf{B}_{\theta,\theta'} &= \{x \in B(0, \Delta_\pi) : |l_\theta(x)| \leq \|\theta - \theta'\|^\kappa\}\ . \end{aligned}$$

Denote by $\mathbb{S}$ the unit sphere $\{x \in \mathbb{R}^d\ /\ \|x\| = 1\}$. Let $u \in [0,1]$ and $tv \in \mathcal{Z}\big(L_{\theta+u(\theta'-\theta)}(\cdot) - L_{\theta+u(\theta'-\theta)}(P^T \cdot)\big) \cap \mathbb{X}$ where $t \in [0, \Delta_\pi]$ and $v \in \mathbb{S}$. Upon noting that for any $\vartheta \in \mathcal{W}_{M+1}$,

$$L_\vartheta(tv) - L_\vartheta(tP^T v) = t\big(q_\vartheta(v)t - l_\vartheta(v)\big)\ , \tag{10.2.23}$$

we consider several cases:

(i) $tv \in \mathsf{B}_{\theta,\theta'}$.

(ii) $tv \notin \mathsf{B}_{\theta,\theta'}$ and $q_{\theta+u(\theta'-\theta)}(v) = 0$. Then, by (10.2.23), $l_{\theta+u(\theta'-\theta)}(tv) = 0$ which implies that $tv \in \mathsf{B}_{\theta,\theta'}$. This yields a contradiction.

(iii) $tv \notin \mathsf{B}_{\theta,\theta'}$ and $q_{\theta+u(\theta'-\theta)}(v) \neq 0$. Then $t \neq 0$ and, by (10.2.23),

$$t = \frac{l_{\theta+u(\theta'-\theta)}(v)}{q_{\theta+u(\theta'-\theta)}(v)} \; . \tag{10.2.24}$$

Since we assumed $t \in [0, \Delta_\pi]$, this ratio is positive. In order to characterize the point tv, additional notations are required. First, note that by Lemma 10, there exists $C_1 > 0$ such that for any $\tilde{\theta} = (\tilde{\mu}, \tilde{\Sigma}) \in \mathcal{W}_{M+1}$,

$$\|\tilde{\theta} - \theta\| \leq h_0 \Rightarrow \|\tilde{\Sigma}^{-1} - \Sigma^{-1}\| \leq C_1 \|\tilde{\Sigma} - \Sigma\| \; .$$

Thus, there exists $C_2 > 0$ such that for any $\tilde{\theta} \in \mathcal{W}_{M+1}$, $\|\tilde{\theta} - \theta\| \leq h_0$, and for any $x \in B(0, \Delta_\pi)$,

$$\begin{aligned} |l_{\tilde{\theta}}(x) - l_\theta(x)| &= 2\left|\mu^T \left[\tilde{\Sigma}^{-1} - \Sigma^{-1}\right](I - P^T)x + (\tilde{\mu} - \mu)^T \tilde{\Sigma}^{-1}(I - P^T)x\right| \\ &\leq C_2 \|\tilde{\theta} - \theta\| \; . \end{aligned} \tag{10.2.25}$$

Note that since $x, \mu \in B(0, \Delta_\pi)$, C_2 does not depend on x and θ. Similarly, there exists $C_3 > 0$ such that for $x \in B(0, \Delta_\pi)$ and $\tilde{\theta} \in \mathcal{W}_{M+1}$ satisfying $\|\tilde{\theta} - \theta\| \leq h_0$,

$$|q_{\tilde{\theta}}(x) - q_\theta(x)| \leq C_3 \|\tilde{\theta} - \theta\| \; . \tag{10.2.26}$$

We can assume without loss of generality that $\bar{h}$ is small enough so that

$$\|\theta - \theta'\| \leq \bar{h} \Rightarrow \|\theta - \theta'\|^\kappa - (C_2 + 2C_3\Delta_\pi) \|\theta - \theta'\| \geq \frac{1}{2} \|\theta - \theta'\|^\kappa \; . \tag{10.2.27}$$

We now distinguish three subcases.

a) $v \in \mathsf{B}_{\theta,\theta'}$.

b) $v \notin \mathsf{B}_{\theta,\theta'}$ and $q_\theta(v) \neq 0$. Since $t \in [0, \Delta_\pi]$, (10.2.24) implies that $|q_{\theta+u(\theta'-\theta)}(v)| \geq |l_{\theta+u(\theta'-\theta)}(v)|/\Delta_\pi$. Since $v \notin \mathsf{B}_{\theta,\theta'}$, $|l_\theta(v)| \geq \|\theta - \theta'\|^\kappa$, and by using (10.2.25)

$$|l_{\theta+u(\theta'-\theta)}| \geq |l_\theta(v)| - \left|l_{\theta+u(\theta'-\theta)} - l_\theta(v)\right| \geq \|\theta - \theta'\|^\kappa - C_2 \|\theta - \theta'\| \; .$$

Hence, it holds that $|q_{\theta+u(\theta'-\theta)}(v)| \geq (\|\theta - \theta'\|^\kappa - C_2\|\theta - \theta'\|)/\Delta_\pi$, and, by (10.2.26), we have $|q_\theta(v)| \geq |q_{\theta+u(\theta'-\theta)}(v)| - C_3\|\theta - \theta'\|$. These inequalities together with (10.2.25) and (10.2.27) lead to

$$\left|t - \frac{l_\theta(v)}{q_\theta(v)}\right| = \left|\frac{l_{\theta+u(\theta'-\theta)}(v)}{q_{\theta+u(\theta'-\theta)}(v)} - \frac{l_\theta(v)}{q_\theta(v)}\right| \leq C_4 \|\theta - \theta'\|^{1-2\kappa},$$

for some $C_4 > 0$.

c) $v \notin \mathsf{B}_{\theta,\theta'}$ and $q_\theta(v) = 0$. Then by (10.2.25) and (10.2.26),

$$t \geq \frac{\|\theta - \theta'\|^\kappa - C_2\|\theta - \theta'\|}{C_3\|\theta - \theta'\|} \geq 2\Delta_\pi \; ,$$

which is in contradiction with the assumption that $t \leq \Delta_\pi$.

As a conclusion, we have just proved that $\mathcal{V}_P$ is included in the union of three sets defined by $\mathsf{B}_{\theta,\theta'}$ (case i), by $\{tv : t \in [0, \Delta_\pi], v \in \mathbb{S} \cap \mathsf{B}_{\theta,\theta'}\}$ (case iiia), and by

$$\left\{tv : v \in \mathbb{S}, v \notin \mathsf{B}_{\theta,\theta'}, q_\theta(v) \neq 0, 0 \le t \le \Delta_\pi, \left|t - \frac{l_\theta(v)}{q_\theta(v)}\right| \le C_4 \|\theta - \theta'\|^{1-2\kappa}\right\}$$

(case iiic). This concludes the first step.

The second step consists in computing an upper bound for the Lebesgue measure of each of these three sets. For simplifying the presentation, we detail the case $d = 2$ and use polar coordinates (ρ, ϕ); the argument remains valid when $d > 2$ using generalized spherical coordinates. Define $t_\theta(\phi) = l_\theta(e^{i\phi})/q_\theta(e^{i\phi})$. Rephrasing the conclusion of the first step, we have $\mathcal{V}_P \subset \bigcup_{\ell=1}^3 \mathcal{V}_P^{(\ell)}$ with

$$\begin{aligned}
&\mathcal{V}_P^{(1)} = \mathsf{B}_{\theta,\theta'} \;, \\
&\mathcal{V}_P^{(2)} = \{(\rho, \phi) \;/\; \rho \in [0, \Delta_\pi], e^{i\phi} \in \mathsf{B}_{\theta,\theta'}\} \;, \\
&\mathcal{V}_P^{(3)} = \{(\rho, \phi) \;/ e^{i\phi} \notin \mathsf{B}_{\theta,\theta'}, q_\theta(e^{i\phi}) \neq 0, 0 \le \rho \le \Delta_\pi, |\rho - t_\theta(\phi)| \le C_4 \|\theta - \theta'\|^{1-2\kappa}\} \;,
\end{aligned}$$

and these sets are Borel sets. By definition of $\mathcal{W}_M$, l_θ is not identically zero and thus

$$\mathrm{Leb}(\mathcal{V}_P^{(1)}) = \mathrm{Leb}(\mathsf{B}_{\theta,\theta'}) \le 2\Delta_\pi \frac{\|\theta - \theta'\|^{1-2\kappa}}{\|2\mu^t \Sigma^{-1}(I - P^T)\|} \le C_5 \|\theta - \theta'\|^{1-2\kappa}$$

for some $C_5 > 0$ as a consequence of Lemma 10. For $\mathcal{V}_P^{(2)}$, note that it is upper bounded by the reunion of the two circular sectors in bold lines in Figure 10.2. This area is easily bounded by the area of the outer rectangle, which is proportional to $\|\theta - \theta'\|^{1-2\kappa}$. Finally,

$$\mathrm{Leb}(\mathcal{V}_P^{(3)}) = \int_0^{2\pi} \left[\frac{\rho^2}{2}\right]_{0 \vee (t_\theta(\phi) - C_4 \|\theta-\theta'\|^{1-2\kappa})}^{\Delta_\pi \wedge (t_\theta(\phi) + C_4 \|\theta-\theta'\|^{1-2\kappa})} \mathbb{1}_{q_\theta(e^{i\phi}) \neq 0} \; d\phi \;.$$

We can assume without loss of generality that $\bar{h}$ is small enough so that $2C_4 \bar{h}^{1-2\kappa} < \Delta_\pi$. Therefore, we can partition $[0, 2\pi] = \mathcal{A} \cup \mathcal{B} \cup \mathcal{C}$, where

$$\begin{aligned}
\mathcal{A} &= \{\phi \in [0, 2\pi] \;/\; t_\theta(\phi) - C_4 \|\theta - \theta'\|^{1-2\kappa} \ge 0 \text{ and } t_\theta(\phi) + C_4 \|\theta - \theta'\|^{1-2\kappa} \le \Delta_\pi\} \;, \\
\mathcal{B} &= \{\phi \in [0, 2\pi] \;/\; t_\theta(\phi) - C_4 \|\theta - \theta'\|^{1-2\kappa} \ge 0 \text{ and } t_\theta(\phi) + C_4 \|\theta - \theta'\|^{1-2\kappa} \ge \Delta_\pi\} \;, \\
\mathcal{C} &= \{\phi \in [0, 2\pi] \;/\; t_\theta(\phi) - C_4 \|\theta - \theta'\|^{1-2\kappa} \le 0 \text{ and } 0 \le t_\theta(\phi) + C_4 \|\theta - \theta'\|^{1-2\kappa} \le \Delta_\pi\} \;.
\end{aligned}$$

This yields

$$\begin{aligned}
\mathrm{Leb}(\mathcal{V}_P^{(3)}) &\le 2C_4 \int_{\mathcal{A}} t_\theta(\phi) \|\theta - \theta'\|^{1-2\kappa} \, d\phi + \frac{1}{2} \int_{\mathcal{B}} \left(\Delta_\pi^2 - \left(t_\theta(\phi) - C_4 \|\theta - \theta'\|^{1-2\kappa}\right)^2\right) d\phi \\
&\quad + \frac{1}{2} \int_{\mathcal{C}} \left(t_\theta(\phi) + C_4 \|\theta - \theta'\|^{1-2\kappa}\right)^2 d\phi \qquad (10.2.28) \\
&\le C_6 \|\theta - \theta'\|^{1-2\kappa} \;, \qquad (10.2.29)
\end{aligned}$$

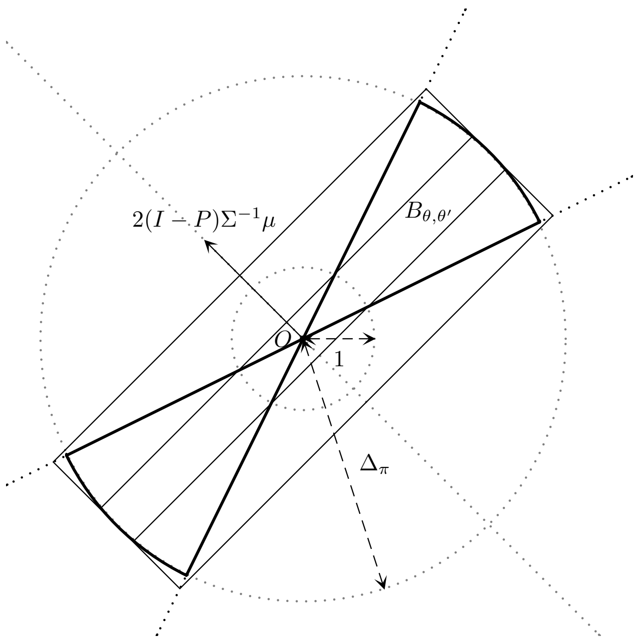


Figure 10.2: Bounding the measure of the set $\mathcal{V}_P^{(2)}$.

for some $C_6 > 0$, since on $\mathcal{A}$, $0 \leq t_\theta(\phi) \leq \Delta_\pi$, on $\mathcal{B}$, $(t_\theta(\phi) - C_4\|\theta-\theta'\|^{1-2\kappa})^2 \geq (\Delta_\pi - 2C_4\|\theta-\theta'\|^{1-2\kappa})^2$, and on $\mathcal{C}$, $|t_\theta(\phi)| \leq C_4\|\theta-\theta'\|^{1-2\kappa}$.

This concludes the proof. □

Lemma 14. *(Regularity in θ of the invariant distribution π_θ)*
Let $M > 0$ and $\kappa \in (0, 1/2)$. Under Assumption 1, there exists $C > 0$ such that for any $\theta \in \mathcal{W}_M$ and $\theta' \in \Theta$,

$$\|\pi_\theta \; - \pi_{\theta'}\|_{\mathrm{TV}} \leq C\|\theta-\theta'\|^{1-2\kappa}.$$

Proof. By definition of the total variation,

$$\begin{aligned}
\|\pi_\theta - \pi_{\theta'}\|_{\mathrm{TV}} &= \sup_{\|f\|_\infty \leq 1} \left| \int f(x)\pi_\theta(x)\, dx - \int f(x)\pi_{\theta'}(x)\, dx \right| \\
&= |\mathcal{P}| \sup_{\|f\|_\infty \leq 1} \left| \int_{V_\theta \setminus V_{\theta'}} f(x)\pi(x)\, dx - \int_{V_{\theta'} \setminus V_\theta} f(x)\pi(x)\, dx \right| \\
&\leq |\mathcal{P}|\big(\pi(V_\theta \setminus V_{\theta'}) + \pi(V_{\theta'} \setminus V_\theta)\big)\,.
\end{aligned}$$

Since

$$V_{\theta'} \setminus V_\theta = V_\theta \setminus \big(V_\theta \cap V_{\theta'}\big)\,, \qquad V_\theta \setminus V_{\theta'} = V_\theta \setminus \big(V_\theta \cap V_{\theta'}\big)\,,$$

it holds that

$$\pi(V_{\theta'} \setminus V_\theta) = \frac{1}{|\mathcal{P}|} - \pi(V_\theta \cap V_{\theta'}) = \pi(V_\theta \setminus V_{\theta'})\,,$$

where we used Lemma 5. Then, by Assumption 1 and Lemma 13, there exists $C > 0$ such that for any $\theta \in \mathcal{W}_M$ and $\theta' \in \Theta$,

$$\|\pi_\theta - \pi_{\theta'}\|_{\mathrm{TV}} \leq 2\|\pi\|_\infty \mathrm{Leb}(V_\theta \setminus V_{\theta'}) \leq C\|\theta - \theta'\|^{1-2\kappa} .$$

□

Lemma 15. *(Regularity in θ of the kernels P_θ)*
Let $M > 0$ and $\kappa \in (0, 1/2)$. Under Assumption 1, there exists $C > 0$ such that for any $\theta \in \mathcal{W}_M$ and $\theta' \in \mathcal{W}_{M+1}$,

$$\|P_\theta(x, \cdot) - P_{\theta'}(x, \cdot)\|_{\mathrm{TV}} \leq C\|\theta - \theta'\|^{1-2\kappa}.$$

Proof. From the definition of the transition kernel P_θ, we have

$$\begin{aligned}
|P_\theta f(x) - P_{\theta'} f(x)| &\leq \left| \int f(y) \Big(\alpha_\theta(x,y) q_\theta(x,y) \mathbb{1}_{V_\theta}(y) - \alpha_{\theta'}(x,y) q_{\theta'}(x,y) \mathbb{1}_{V_{\theta'}}(y) \Big) \, dy \right| \\
&\quad + |f(x)| \left| \int \Big(\alpha_{\theta'}(x,y) q_{\theta'}(x,y) \mathbb{1}_{V_{\theta'}}(y) - \alpha_\theta(x,y) q_\theta(x,y) \mathbb{1}_{V_\theta}(y) \Big) \, dy \right| \\
&\leq 2\|f\|_\infty \int \Big| \alpha_\theta(x,y) q_\theta(x,y) \mathbb{1}_{V_\theta}(y) - \alpha_{\theta'}(x,y) q_{\theta'}(x,y) \mathbb{1}_{V_{\theta'}}(y) \Big| \, dy \\
&= 2\|f\|_\infty \sum_{i=1}^4 \Delta^i_{\theta,\theta'}(x) , \qquad (10.2.30)
\end{aligned}$$

where

$$\begin{aligned}
\Delta^1_{\theta,\theta'}(x) &= \int_{\mathcal{A}_\theta(x) \cap \mathcal{A}_{\theta'}(x)} \Big| \alpha_\theta(x,y) q_\theta(x,y) \mathbb{1}_{V_\theta}(y) - \alpha_{\theta'}(x,y) q_{\theta'}(x,y) \mathbb{1}_{V_{\theta'}}(y) \Big| \, dy , \\
\Delta^2_{\theta,\theta'}(x) &= \int_{\mathcal{R}_\theta(x) \cap \mathcal{R}_{\theta'}(x)} \Big| \alpha_\theta(x,y) q_\theta(x,y) \mathbb{1}_{V_\theta}(y) - \alpha_{\theta'}(x,y) q_{\theta'}(x,y) \mathbb{1}_{V_{\theta'}}(y) \Big| \, dy , \\
\Delta^3_{\theta,\theta'}(x) &= \int_{\mathcal{A}_\theta(x) \cap \mathcal{R}_{\theta'}(x)} \Big| \alpha_\theta(x,y) q_\theta(x,y) \mathbb{1}_{V_\theta}(y) - \alpha_{\theta'}(x,y) q_{\theta'}(x,y) \mathbb{1}_{V_{\theta'}}(y) \Big| \, dy , \\
\Delta^4_{\theta,\theta'}(x) &= \int_{\mathcal{R}_\theta(x) \cap \mathcal{A}_{\theta'}(x)} \Big| \alpha_\theta(x,y) q_\theta(x,y) \mathbb{1}_{V_\theta}(y) - \alpha_{\theta'}(x,y) q_{\theta'}(x,y) \mathbb{1}_{V_{\theta'}}(y) \Big| \, dy ,
\end{aligned}$$

and

$$\mathcal{A}_\theta(x) = \{y : \alpha_\theta(x,y) = 1\} , \qquad \mathcal{R}_\theta(x) = \{y : \alpha_\theta(x,y) < 1\} .$$

We now upper bound each term in turn.

$$\begin{aligned}
\Delta^1_{\theta,\theta'}(x) &= \int_{\mathcal{A}_\theta(x) \cap \mathcal{A}_{\theta'}(x)} \Big| \sum_{Q \in \mathcal{P}} \Big(\mathbb{1}_{V_\theta}(y) \mathcal{N}(Qy|x, \Sigma) - \mathbb{1}_{V_{\theta'}}(y) \mathcal{N}(Qy|x, \Sigma') \Big) \Big| \, dy \\
&\leq \int \big| \mathbb{1}_{V_\theta}(y) - \mathbb{1}_{V_{\theta'}}(y) \big| \sum_{Q \in \mathcal{P}} \mathcal{N}(Qy|x, \Sigma) + \\
&\qquad \mathbb{1}_{V_{\theta'}}(y) \sum_{Q \in \mathcal{P}} \big| \mathcal{N}(Qy|x, \Sigma) - \mathcal{N}(Qy|x, \Sigma') \big| \, dy . \qquad (10.2.31)
\end{aligned}$$

By Lemma 10, there exist $a, b > 0$ such that for any $\theta \in \mathcal{W}_{M+1}$, $m, z \in \mathbb{X}$, and $Q \in \mathcal{P}$, we have

$$a \leq \mathcal{N}(Qz|m, c\Sigma) \leq b\,, \tag{10.2.32}$$

so that the first term in the RHS of (10.2.31) is bounded by

$$\begin{aligned}\int \left|\mathbb{1}_{V_\theta}(y) - \mathbb{1}_{V_{\theta'}}(y)\right| \sum_{Q\in\mathcal{P}} \mathcal{N}(Qy|x,\Sigma)\,dy &\leq |\mathcal{P}|b \int \left|\mathbb{1}_{V_\theta}(y) - \mathbb{1}_{V_{\theta'}}(y)\right|\,dy \\ &= |\mathcal{P}|b \int \left(\mathbb{1}_{V_\theta\setminus V_{\theta'}}(y) + \mathbb{1}_{V_{\theta'}\setminus V_\theta}(y)\right)\,dy \\ &\leq C\|\theta-\theta'\|^{1-2\kappa}\,,\end{aligned}$$

where we used Lemma 13. Let us now consider the second term of the right-hand side of (10.2.31). Using the uniform continuity of w on $\mathcal{W}_{M+1}$ (see Lemma 10), there exists $\bar{h}$ small enough such that

$$\theta \in \mathcal{W}_M, \|h\| < \bar{h} \Rightarrow \theta + h \in \mathcal{W}_{M+1}. \tag{10.2.33}$$

For any $\theta \in \mathcal{W}_M$, $\theta' \in \mathcal{W}_{M+1}$ such that $\|\theta - \theta'\| \geq \bar{h}$, there exists C_1 such that

$$\sum_{Q\in\mathcal{P}} \left|\mathcal{N}(Qy|x,\Sigma) - \mathcal{N}(Qy|x,\Sigma')\right|\,dy \leq C_1\|\theta-\theta'\|^{1-2\kappa}\,.$$

Assume now that $\theta \in \mathcal{W}_M$, $\theta' \in \mathcal{W}_{M+1}$ and $\|\theta - \theta'\| < \bar{h}$. Denote by

$$\Sigma_t = (1-t)\Sigma + t\Sigma'\,. \tag{10.2.34}$$

By (10.2.33) and (10.2.7b), Σ_t^{-1} exists and $\sup_{t\leq 1,\theta\in\mathcal{W}_M,\theta'\in\mathcal{W}_{M+1}} \|\Sigma_t^{-1}\| < \infty$. We can then write

$$\begin{aligned}\left|\mathcal{N}(Qy|x,\Sigma) - \mathcal{N}(Qy|x,\Sigma')\right| &= \int_0^1 \mathcal{N}(Qy|x,\Sigma_t)\left|\frac{d}{dt}\log\mathcal{N}(Qy|x,\Sigma_t)\right|\,dt \\ &\leq b\int_0^1 \left|\frac{d}{dt}\log\mathcal{N}(Qy|x,\Sigma_t)\right|\,dt\,. \end{aligned} \tag{10.2.35}$$

In addition, by Assumption 1, there exists C_2 such that

$$\left|\frac{d}{dt}\log\mathcal{N}(Qy|x,\Sigma_t)\right| = \left|(x-Qy)^T\Sigma_t^{-1}(\Sigma'-\Sigma)\Sigma_t^{-1}(x-Qy)\right| \leq C_2\|\theta-\theta'\|\,. \tag{10.2.36}$$

We thus have proved that

$$\left[\theta\in\mathcal{W}_M\;,\theta'\in\mathcal{W}_{M+1}\;,\|\theta-\theta'\|<\bar{h}\right] \Longrightarrow \left|\mathcal{N}(Qy|x,\Sigma)-\mathcal{N}(Qy|x,\Sigma')\right| \leq C\|\theta-\theta'\|\,.$$

Therefore, it is established that $\|\Delta^1_{\theta,\theta'}\|_\infty \leq C\|\theta-\theta'\|^{1-2\kappa}$.

Let us consider the second term $\Delta^2_{\theta,\theta'}(x)$ in the RHS of (10.2.30). Note first that if $x \in \mathbb{X}$ and $y \in \mathcal{R}_\theta(x) \cap \mathcal{R}_{\theta'}(x)$, then by (10.2.32), $\pi(y)/\pi(x) \leq b/a$, so

$$\begin{aligned}
\Delta^2_{\theta,\theta'}(x) &= \int_{\mathcal{R}_\theta(x)\cap\mathcal{R}_{\theta'}(x)} \frac{\pi(y)}{\pi(x)} \left| \sum_{Q\in\mathcal{P}} \Big(\mathbb{1}_{V_\theta}(y)\mathcal{N}(Qx|y,\Sigma) - \mathbb{1}_{V_{\theta'}}(y)\mathcal{N}(Qx|y,\Sigma') \Big) \right| dy \\
&\leq \frac{b}{a} \int_{\mathcal{R}_\theta(x)\cap\mathcal{R}_{\theta'}(x)} \left| \sum_{Q\in\mathcal{P}} \Big(\mathbb{1}_{V_\theta}(y)\mathcal{N}(Qx|y,\Sigma) - \mathbb{1}_{V_{\theta'}}(y)\mathcal{N}(Qx|y,\Sigma') \Big) \right| dy \;.
\end{aligned}$$

Therefore, repeating the above discussion for the bound of $\Delta^1_{\theta,\theta'}(x)$, it is established that $\|\Delta^2_{\theta,\theta'}\|_\infty \leq C\|\theta - \theta'\|^{1-2\kappa}$.

To deal with $\Delta^3_{\theta,\theta'}(x)$, first observe that there exists $C > 0$ such that for any $\theta \in \mathcal{W}_M$, $\theta' \in \mathcal{W}_{M+1}$, and $x, y \in \mathbb{X}$, we have

$$\left| \frac{q_\theta(y,x)}{q_\theta(x,y)} - \frac{q_{\theta'}(y,x)}{q_{\theta'}(x,y)} \right| \leq C\|\theta - \theta'\| \;, \tag{10.2.37}$$

because of (10.1.6), (10.2.32), and the above discussion for the upper bound of $\Delta^1_{\theta,\theta'}(x)$. Now let $y \in \mathcal{A}_\theta(x) \cap \mathcal{R}_{\theta'}(x)$, then we have

$$\frac{\pi(y)q_{\theta'}(y,x)}{\pi(x)q_{\theta'}(x,y)} \leq 1 \leq \frac{\pi(y)q_\theta(y,x)}{\pi(x)q_\theta(x,y)} \;,$$

which, combined with (10.2.37), yields

$$1 - C\frac{\pi(y)}{\pi(x)}\|\theta - \theta'\| \leq \frac{\pi(y)q_{\theta'}(y,x)}{\pi(x)q_{\theta'}(x,y)} \leq 1 \;.$$

Thus,

$$\begin{aligned}
\Delta^3_{\theta,\theta'}(x) &= \int_{\mathcal{A}_\theta(x)\cap\mathcal{R}_{\theta'}(x)} \left| q_\theta(x,y)\mathbb{1}_{V_\theta}(y) - \frac{\pi(y)q_{\theta'}(y,x)}{\pi(x)q_{\theta'}(x,y)} q_{\theta'}(x,y)\mathbb{1}_{V_{\theta'}}(y) \right| dy \\
&\leq \int \Big(\big| q_\theta(x,y)\mathbb{1}_{V_\theta}(y) - q_{\theta'}(x,y)\mathbb{1}_{V_{\theta'}}(y) \big| \vee \cdots \\
&\qquad \left| q_\theta(x,y)\mathbb{1}_{V_\theta}(y) - q_{\theta'}(x,y)\mathbb{1}_{V_{\theta'}}(y) + C\frac{\pi(y)}{\pi(x)}\|\theta - \theta'\| q_{\theta'}(x,y)\mathbb{1}_{V_{\theta'}}(y) \right| \Big) \, dy \;.
\end{aligned}$$

Therefore, it is established that $\|\Delta^3_{\theta,\theta'}\|_\infty \leq C\|\theta - \theta'\|^{1-2\kappa}$.

The upper bound of $\Delta^4_{\theta,\theta'}(x)$ is similar and thus its proof is omitted. □

Lemma 16. *(Regularity in θ of the solution of the Poisson equation)*
Let $M > 0$ and $\kappa \in (0, 1/2)$. Under Assumption 1, there exists $C > 0$ such that for any $\theta \in \mathcal{W}_M$ and $\theta' \in \mathcal{W}_{M+1}$,

$$\|P_\theta \hat{H}_\theta - P_{\theta'} \hat{H}_{\theta'}\|_\infty \leq C\|\theta - \theta'\|^{1-2\kappa}.$$

Proof. We recall the following result, proved in [59, Lemma 5.5, page 24]: there exists $C > 0$ such that for any $\theta \in \mathcal{W}_M$, $\theta' \in \mathcal{W}_{M+1}$, and $x \in \mathbb{X}$,

$$\|P_\theta \hat{H}_\theta - P_{\theta'} \hat{H}_{\theta'}\|_\infty \leq C\|H(\cdot,\theta) - H(\cdot,\theta')\|_\infty + C \sup_{\theta \in \mathcal{W}_M} \|H(\cdot,\theta)\|_\infty \big\{ \|\pi_\theta - \pi_{\theta'}\|_{\mathrm{TV}} + \sup_{x \in \mathbb{X}} \|P_\theta(x,\cdot) - P_{\theta'}(x,\cdot)\|_{\mathrm{TV}} \big\} . \tag{10.2.38}$$

Here $\sup_{\theta \in \mathcal{W}_M} \|H(\cdot,\theta)\|_\infty$ is finite by Lemma 10. Now, by Lemma 10 again, there exists $C > 0$ such that for any $\theta \in \mathcal{W}_M$ and $\theta' \in \mathcal{W}_{M+1}$,

$$\|H(\cdot,\theta) - H(\cdot,\theta')\|_\infty \quad \leq \quad C\|\theta - \theta'\|.$$

The upper bounds for the two last terms in the RHS of (10.2.38) result from Lemmas 14 and 15, respectively. □

10.2.6 Proof of Theorem 2

The proof is prefaced with two lemmas.

Lemma 17. *Let $(\gamma_t)_{t>0}$ be a sequence such that $\sum_t \gamma_t^2 < \infty$, $\sum_t |\gamma_{t+1} - \gamma_t| < \infty$, and $\sum_t \gamma_t^{2(1-\kappa)} < \infty$ for some $\kappa \in (0, 1/2)$. Denote by ψ_t the value of the projection counter at the end of iteration t in Figure 10.1. Let also $(\theta_t, X_t)_{t\geq 0}$ be the sequence generated by the stable AMOR algorithm in Figure 10.1. Under Assumptions 1 and 2, for any $M > 0$,*

$$\lim_{L\to+\infty} \sup_{\ell \geq 1} \left\| \left(\prod_{k=L}^{L+\ell} \mathbb{1}_{\theta_k \in \mathcal{W}_M} \mathbb{1}_{\psi_{k+1}=\psi_k} \right) \sum_{k=L}^{L+\ell} \gamma_{k+1} \big(H(X_{k+1},\theta_k) - h(\theta_k) \big) \right\| = 0 \qquad w.p.1 \,, \tag{10.2.39}$$

where H, h, w, and $\mathcal{W}_M$ are given by (10.1.2), (10.1.10), (10.1.11), and (10.2.6), respectively.

Proof. Let $M > 0$. By uniform continuity of w on $\mathcal{W}_{M+1}$, let $L(M)$ be large enough so that

$$L \geq L(M), \theta \in \mathcal{W}_M \Longrightarrow \forall x \in \mathbb{X}\,, \theta + \gamma_{L+1} H(x,\theta) \in \mathcal{W}_{M+1} \,. \tag{10.2.40}$$

Let $L \geq L(M)$ and denote by

$$\mathbb{I}_{L,\ell} = \prod_{k=L}^{L+\ell} \mathbb{1}_{\theta_k \in \mathcal{W}_M} \mathbb{1}_{\psi_{k+1}=\psi_k} \,.$$

For any $\theta \in \mathcal{W}_M$, Lemma 12 implies that there exists a function $\hat{H}_\theta$ such that

$$\hat{H}_\theta - P_\theta \hat{H}_\theta = H(\cdot,\theta) - \pi_\theta(H(\cdot,\theta)) \qquad \text{and} \qquad \sup_{x \in \mathbb{X}, \theta \in \mathcal{W}_M} \|\hat{H}_\theta(x)\| < \infty \,.$$

Therefore, for $\ell + L \geq i \geq L \geq 0$, we have

$$\mathbb{I}_{L,\ell}\big(H(X_{i+1}, \theta_i) - h(\theta_i)\big) = \mathbb{I}_{L,\ell}\big(M_{i+1} + R^{(1)}_{i+1} + R^{(2)}_{i+1}\big) ,$$

where

$$\begin{aligned} M_{i+1} &= \left(\hat{H}_{\theta_i}(X_{i+1}) - P_{\theta_i}\hat{H}_{\theta_i}(X_i)\right) , && (10.2.41) \\ R^{(1)}_{i+1} &= P_{\theta_i}\hat{H}_{\theta_i}(X_i) - P_{\theta_{i+1}}\hat{H}_{\theta_{i+1}}(X_{i+1}) , && (10.2.42) \\ R^{(2)}_{i+1} & = P_{\theta_{i+1}}\hat{H}_{\theta_{i+1}}(X_{i+1}) - P_{\theta_i}\hat{H}_{\theta_i}(X_{i+1}) . && (10.2.43) \end{aligned}$$

First note that

$$\mathbb{I}_{L,\ell}\sum_{i=L}^{L+\ell}\gamma_{i+1}M_{i+1} = \mathbb{I}_{L,\ell}\left(\sum_{i=0}^{L+\ell}\gamma_{i+1}\mathbb{I}_{i,0}M_{i+1} - \sum_{i=0}^{L-1}\gamma_{i+1}\mathbb{I}_{i,0}M_{i+1}\right) . \qquad (10.2.44)$$

By Lemma 12, $\{\mathbb{I}_{i,0}M_{i+1}\}_i$ is a martingale-increment. Therefore, by [70], a sufficient condition for $\sum_{i\geq 0}\gamma_{i+1}\mathbb{I}_{i,0}M_{i+1}$ to converge to zero is

$$\sum_{i\geq 0}\gamma^2_{i+1}\mathbb{E}\left(\|\hat{H}_{\theta_i}(X_{i+1}) - P_{\theta_i}\hat{H}_{\theta_i}(X_i)\|^2\ \mathbb{I}_{i,0}\right) < \infty . \qquad (10.2.45)$$

By the parallelogram identity and Hölder's inequality,

$$\|\hat{H}_{\theta_i}(X_{i+1}) - P_{\theta_i}\hat{H}_{\theta_i}(X_i)\|^2\ \mathbb{I}_{i,0} \leq 4 \sup_{x\in\mathbb{X},\theta\in\mathcal{W}_M}\|\hat{H}_\theta(x)\|^2 .$$

Eqn. (10.2.45) then holds since $\sum_t \gamma_t^2 < \infty$. By (10.2.44), we obtain that

$$\lim_{L\to\infty}\sup_{\ell\geq 1}\left|\mathbb{I}_{L,\ell}\sum_{k=L}^{L+\ell}\gamma_{k+1}M_{k+1}\right| = 0 \quad \text{w.p. } 1 .$$

Let us now consider the term $R^{(1)}_{i+1}$ defined in (10.2.42). Summing by parts, we get

$$\begin{aligned} \mathbb{I}_{L,\ell}\sum_{i=L}^{L+\ell}\gamma_{i+1}R^{(1)}_{i+1} &= \mathbb{I}_{L,\ell}\ \gamma_{L+1}P_{\theta_L}\hat{H}_{\theta_L}(X_L) + \mathbb{I}_{L,\ell}\sum_{i=L+1}^{L+\ell}(\gamma_{i+1} - \gamma_i)P_{\theta_i}\hat{H}_{\theta_i}(X_i) \\ &\quad -\mathbb{I}_{L,\ell}\gamma_{L+\ell+1}P_{\theta_{L+\ell+1}}\hat{H}_{\theta_{L+\ell+1}}(X_{L+\ell+1}) . \end{aligned}$$

Since $\sup_{x\in\mathbb{X},\theta\in\mathcal{W}_M}\|\hat{H}_\theta(x)\| < \infty$, there exists a constant C such that the RHS is upper bounded by $C\left(|\gamma_{L+1}| + \sum_{i\geq\ell+1}|\gamma_{i+1} - \gamma_i| + |\gamma_{L+\ell+1}|\right)$. Under the stated assumptions, this upper bound yields

$$\lim_{L\to\infty}\sup_{\ell\geq 1}\left|\mathbb{I}_{L,\ell}\sum_{i=L}^{L+\ell}\gamma_{i+1}R^{(1)}_{i+1}\right| = 0 ,$$

with probability 1.

Finally, let us consider the term $R^{(2)}_{i+1}$ defined in (10.2.43). By (10.2.40), Lemma 16, and since on the event $\{\psi_{k+1} = \psi_k\}$, we have $\theta_{k+1} = \theta_k + \gamma_{k+1} H(X_{k+1}, \theta_k)$, we obtain

$$\begin{aligned} \mathbb{I}_{L,\ell} \left| \sum_{i=L}^{L+\ell} \gamma_{i+1} R^{(2)}_{i+1} \right| &\leq \mathbb{I}_{L,\ell} \sum_{i=L}^{L+\ell} \gamma_{i+1} \left\| P_{\theta_{i+1}} \hat{H}_{\theta_{i+1}} - P_{\theta_i} \hat{H}_{\theta_i} \right\|_\infty \\ &\leq C\, \mathbb{I}_{L,\ell} \sum_{i=L}^{L+\ell} \gamma_{i+1} \|\theta_{i+1} - \theta_i\|^{1-2\kappa} \leq C' \sum_{i=L}^{L+\ell} \gamma_{i+1}^{2(1-\kappa)} . \end{aligned}$$

This concludes the proof.

□

Lemma 18. *Let $M \in (0, M_\star)$ and set*

$$\Gamma^M_{M_\star} = \{\theta \in \Theta : M_\star \leq w(\theta) \leq M\} , \qquad \iota = \inf_{\theta \in \Gamma^M_{M_\star}} |\langle \nabla w(\theta), h(\theta) \rangle| .$$

Under Assumptions 1 and 2, there exist $\delta \in (0, \iota)$ and $\lambda, \beta > 0$ such that

(A) $u \in \mathcal{W}_{M_\star}, 0 \leq \gamma \leq \lambda, \|\xi\| \leq \beta \Rightarrow w(u + \gamma(u) + \gamma\xi) \leq M$, and

(B) $u \in \Gamma^M_{M_\star}, 0 \leq \gamma \leq \lambda, \|\xi\| \leq \beta \Rightarrow w(u + \gamma(u) + \gamma\xi) < w(u) - \gamma\delta$.

Proof. Define $u' = u + \gamma(u) + \gamma\xi$.

(A) Let $u \in \mathcal{W}_M$. Since w is continuous on Θ and the level set $\mathcal{W}_M$ is a compact subset of Θ (see Lemma 10), there exists $\eta > 0$ such that for any $u \in \mathcal{W}_M$ and any u' satisfying $\|u' - u\| \leq \eta$, $u' \in \mathcal{W}_{M+1}$. Therefore, since

$$\|u - u'\| \leq \lambda(\max_{\mathcal{W}_M} \|\| + \beta), \tag{10.2.46}$$

there exists $\lambda_1, \beta_1 > 0$ such that for any $0 \leq \gamma \leq \lambda_1$ and any $\|\xi\| \leq \beta_1$, $u' \in \mathcal{W}_{M+1}$ (note that $\max_{\mathcal{W}_M} \|\| < \infty$ by Lemma 10).

Since w is continuous on the compact set $\mathcal{W}_{M+1}$ (see Lemma 10), it is uniformly continuous (u.c.) on $\mathcal{W}_{M+1}$. Then we can choose $\lambda_2, \beta_2 > 0$ (smaller than λ_1, β_1) such that

$$\forall u \in \mathcal{W}_{M_\star} , \forall \gamma \leq \lambda_2, \|\xi\| \leq \beta_2 , \quad \left| w(u) - w(u + \gamma(u) + \gamma\xi) \right| \leq M - M_\star . \tag{10.2.47}$$

This concludes the proof of (A).

(B) Let $u \in \Gamma^M_{M_\star}$. Following the same lines as in the proof of (10.2.47), there exist $\lambda_1, \beta_1 > 0$ such that for any $0 \leq \gamma \leq \lambda_1$ and $\|\xi\| \leq \beta_1$, $[u, u'] \subset \mathcal{W}_{M+1}$. By Lemma 10, this implies that w is continuously differentiable on (u, u'). We write

$$\begin{aligned} \left| \langle \nabla w(u), (u) \rangle - \langle \nabla w(u'), (u) + \xi \rangle \right| &= \left| \langle \nabla w(u), (u) \rangle - \langle \nabla w(u'), (u') \rangle \right. \\ &\quad \left. + \langle \nabla w(u'), (u') - (u) - \xi \rangle \right| . \end{aligned}$$

By Lemma 10, $\varphi : u \mapsto \langle \nabla w(u), h(u) \rangle$ is continuous and negative on the compact set $\Gamma_{M_\star}^{M}$, so there exists $\varepsilon \in (0, \iota)$ such that $\langle \nabla w(u), h(u) \rangle \leq -\varepsilon$ on $\Gamma_{M_\star}^{M}$. Furthermore, φ is u.c. on $\mathcal{W}_{M+1}$, and, for any $\varepsilon' > 0$, we can thus take β_2 and λ_2 small enough so that for any $0 \leq \gamma \leq \lambda_2$ and $\|\xi\| \leq \beta_2$, $|\varphi(u) - \varphi(u')| \leq \varepsilon'/2$. Therefore

$$\left| \langle \nabla w(u), (u) \rangle - \langle \nabla w(u'), (u) + \xi \rangle \right| \leq \varepsilon'/2 + \big(\|(u) - (u')\| + \beta_2 \big) \max_{\mathcal{W}_{M+1}} \|\nabla w\| \, .$$

Since $x \mapsto \|\nabla w(x)\|$ is continuous on the compact set $\mathcal{W}_{M+1}$, $\max_{\mathcal{W}_{M+1}} \|\nabla w\|$ is finite. As is u.c. on $\mathcal{W}_{M+1}$, one can pick λ_2, β_2 small enough so that

$$\forall u \in \Gamma_{M_\star}^{M} \, , \forall \gamma \leq \lambda_2, \|\xi\| \leq \beta_2 \text{ , and } \left| \langle \nabla w(u), (u) \rangle - \langle \nabla w(u'), (u) + \xi \rangle \right| \leq \varepsilon' \, .$$

Finally, applying Taylor's formula, we get

$$\begin{aligned} w(u') - w(u) &= \int_0^1 \Big\langle \nabla w(u + t\gamma((u) + \xi)), \gamma((u) + \xi) \Big\rangle dt \\ &= \gamma \varphi(u) + \gamma \int_0^1 \Big(\big\langle \nabla w(u + t\gamma((u) + \xi)), (u) + \xi \big\rangle - \big\langle \nabla w(u), (u) \big\rangle \Big) dt \\ &\leq -\gamma\varepsilon + \gamma\varepsilon'. \end{aligned}$$

Since ε' is arbitrary, this yields (B). □

Proof of Item 1 in Theorem 2. Let $M > M_\star$ and q (depending upon M) be such that (see Remark 11)

$$\mathcal{W}_M \subset \mathcal{W}_{M+2} \subseteq \mathcal{K}_{\delta_q} \, ; \tag{10.2.48}$$

and $\theta_0 \in \mathcal{W}_M$. Let λ, β be given by Lemma 18. By Lemma 10, w and h are uniformly continuous on $\mathcal{W}_{M+1}$, and there exists $\eta > 0$ such that

$$x \in \mathcal{W}_M, \|x - y\| < \eta \Longrightarrow |w(x) - w(y)| < 1 \text{ and } \|h(x) - h(y)\| < \beta \, . \tag{10.2.49}$$

By Lemma 17, there exists an almost surely finite r.v. N such that w.p.1.,

$$n \geq N \Rightarrow \gamma_n \left(1 + \sup_{x \in \mathbb{X}, \theta \in \mathcal{W}_M} \|H(x, \theta)\| \right) < \lambda \wedge \eta, \text{ and} \tag{10.2.50}$$

$$\sup_{\ell \geq 1} \left(\prod_{i=N}^{N+\ell} \mathbb{1}_{\theta_i \in \mathcal{W}_{M+1}} \mathbb{1}_{\psi_{i+1} = \psi_i} \right) \left\| \sum_{i=N}^{N+\ell} \gamma_{i+1} \left(H(X_{i+1}, \theta_i) - h(\theta_i) \right) \right\| < \eta \tag{10.2.51}$$

The proof is by contradiction. Denote by ψ_t the number of projections at the end of iteration t. We assume that $\mathbb{P}(\lim_t \psi_t = +\infty) > 0$. We can assume without loss of generality that on the set $\{\lim_t \psi_t = +\infty\}$

$$w(\theta_N) \leq M \, , \qquad \psi_N \geq q \, .$$

Define the sequence $(\theta'_{N+k})_{k \geq 0}$ as

$$\theta'_N = \theta_N \quad \text{and} \quad \theta'_{N+k+1} = \theta'_{N+k} + \gamma_{N+k+1}(\theta_{N+k}) \, .$$

We prove by induction on k that for any $k \geq 0$, on the set $\{\lim_t \psi_t = +\infty\}$,

$$\theta'_{N+k} \in \mathcal{W}_M \;, \qquad \theta_{N+k} \in \mathcal{W}_{M+1} \;, \qquad \|\theta'_{N+k} - \theta_{N+k}\| < \eta \;, \qquad \psi_{N+k+1} = \psi_{N+k} \;.$$

The case $k = 0$ is trivial since $\theta'_N = \theta_N \in \mathcal{W}_M$ and by using (10.2.49), (10.2.50), and (10.2.48) on the set $\{\lim_t \psi_t = +\infty\}$. Assume this property holds for $k \in \{0, 1, ..., \ell\}$. Then we have

$$\theta'_{N+\ell+1} = \theta'_{N+\ell} + \gamma_{N+\ell+1}(\theta'_{N+\ell}) + \gamma_{N+\ell+1}\big((\theta_{N+\ell}) - (\theta'_{N+\ell})\big).$$

Since $\|\theta'_{N+\ell} - \theta_{N+\ell}\| < \eta$ and $\theta'_{N+\ell}$ is in $\mathcal{W}_M$, we have $\|(\theta'_{N+\ell}) - (\theta_{N+\ell})\| < \beta$. Since $\gamma_{N+\ell+1} < \lambda$ by (10.2.50), we can apply Lemma 18 to obtain $\theta'_{N+\ell+1} \in \mathcal{W}_M$. In addition,

$$\begin{aligned}\theta'_{N+\ell+1} - \theta_{N+\ell+1} &= \sum_{i=N}^{N+\ell} \gamma_{i+1}\big(H(X_{i+1}, \theta_i) - h(\theta_i)\big)\mathbb{1}_{\psi_{i+1}=\psi_i} + \sum_{i=N}^{N+\ell} \big(\gamma_{i+1} h(\theta_i) + \theta_i - \theta_0\big)\mathbb{1}_{\psi_{i+1}\neq\psi_i} \\ &= \left(\prod_{i=N}^{N+\ell} \mathbb{1}_{\theta_i \in \mathcal{W}_{M+1}}\right) \sum_{i=N}^{N+\ell} \gamma_{i+1}\big(H(X_{i+1}, \theta_i) - h(\theta_i)\big)\mathbb{1}_{\psi_{i+1}=\psi_i} \;,\end{aligned}$$

where we used the induction assumption in the last equality. From (10.2.49) and (10.2.51), this yields $\|\theta'_{N+\ell+1} - \theta_{N+\ell+1}\| < \eta$ and $w(\theta_{N+\ell+1}) \leq M + 1$. Finally by (10.2.49), Eqs. (10.2.50) and (10.2.48) imply that on the set $\{\lim_t \psi_t = +\infty\}$

$$\theta_{N+\ell} + \gamma_{N+\ell+1} H(X_{N+\ell+1}, \theta_{N+\ell}) \in \mathcal{W}_{M+2} \subset \mathcal{K}_{\psi_{N+\ell}} \;,$$

that is, $\psi_{N+\ell+1} = \psi_{N+\ell}$. This concludes the induction.

As a consequence of this induction, we have $\psi_{N+\ell} = \psi_N$ for any $\ell \geq 0$ on the set $\{\lim_t \psi_t = +\infty\}$ which is a contradiction.

Proof of Item 2 in Theorem 2. The proof is along the same lines as the proof of Theorem 2.3 of [6, page 5], and is thus omitted.

10.2.7 Proof of Theorem 3

The proof consists in checking the conditions of [59, Corollary 2.8]. Let f be a measurable bounded function.

By Lemma 12, *(i)* there exists a measurable function $\hat{f}_\theta$ such that $\hat{f}_\theta - P_\theta \hat{f}_\theta = f - \pi_\theta f$; and *(ii)* for any compact set $\mathcal{W}_M$, there exists L (depending upon M) such that

$$\forall \theta \in \mathcal{W}_M, x \in \mathbb{X}, \; |\hat{f}_\theta(x)| \leq L \;.$$

By Theorem 2, $\mathbb{P}(\Omega_M) \uparrow 1$ when M tends to infinity where

$$\Omega_M = \bigcap_{t \geq 0} \{\theta_t \in \mathcal{W}_M\} \;.$$

Therefore, in order to apply [59, Corollary 2.8], we only have to prove that almost surely,

$$\sum_k k^{-1} \sup_{x\in\mathbb{X}} \|P_{\theta_k}(x,\cdot) - P_{\theta_{k-1}}(x,\cdot)\|_{\mathrm{TV}} \mathbb{1}_{\Omega_M} < \infty \;, \tag{10.2.52}$$

$$\lim_t \pi_{\theta_t}(f) \mathbb{1}_{\Omega_M} = \pi_{\theta^\star}(f)\mathbb{1}_{\Omega_M} \;. \tag{10.2.53}$$

By Lemma 15, there exists C and $\kappa \in (0, 1/2)$ such that

$$\sup_{x\in\mathbb{X}} \|P_{\theta_k}(x,\cdot) - P_{\theta_{k-1}}(x,\cdot)\|_{\mathrm{TV}} \mathbb{1}_{\Omega_M} \le C \left\|\theta_k - \theta_{k-1}\right\|^{1-2\kappa} \;.$$

In addition, by Theorem 2, there exists a random variable K, almost-surely finite, such that for any $k \ge K$,

$$\|\theta_k - \theta_{k-1}\| \mathbb{1}_{\Omega_M} \le \gamma_k \sup_{\theta\in\mathcal{W}_M, x\in\mathbb{X}} |H(x,\theta)| \;.$$

This yields

$$\sum_{k\ge K} k^{-1} \sup_{x\in\mathbb{X}} \|P_{\theta_k}(x,\cdot) - P_{\theta_{k-1}}(x,\cdot)\|_{\mathrm{TV}} \mathbb{1}_{\Omega_M} \le C \sum_{k\ge K} k^{-1}\gamma_k^{1-2\kappa} \;,$$

for some constant $C > 0$. This concludes the proof of (10.2.52). The limit (10.2.53) is a consequence of Lemma 14.

10.2.8 Proof of Theorem 4

Let f be a measurable function such that $\|f\|_\infty \le 1$ and set

$$I_t(f) = \left|\mathbb{E}[f(X_t)\mathbb{1}_B] - \pi_{\theta^\star}(f)\mathbb{P}(B)\right| = \left|\mathbb{E}[\big(f(X_t) - \pi_{\theta^\star}(f)\big)\mathbb{1}_B]\right| \;.$$

Let $\varepsilon > 0$. We prove that there exists T_ε such that for all $t \ge T_\varepsilon$, $\sup_{\{f:\|f\|_\infty\le 1\}} I_t(f) \le 4\varepsilon$. Choose $\kappa \in (0, 1/2)$ and $\delta > 0$ such that

$$C_{M_\star+1}\delta^{1-2\kappa} \le \varepsilon \;, \tag{10.2.54}$$

where $M_\star$ and $C_{M_\star}$ are defined in Assumption 2 and in Lemma 14, respectively. Choose r_ε such that

$$2(1-\rho_{M_\star+1})^{r_\varepsilon} \le \varepsilon \;, \tag{10.2.55}$$

where $\rho_{M_\star+1}$ is defined in Lemma 12. By uniform continuity of w on $\mathcal{W}_{M_\star+2}$, assume finally δ is small enough that

$$\theta \in \mathcal{W}_{M_\star+1}, \theta' \in \Theta, \|\theta - \theta'\| \le \delta \Rightarrow |w(\theta) - w(\theta')| \le \frac{1}{r_\varepsilon + 1} \;. \tag{10.2.56}$$

There exists T^1_ε such that for any $t \ge T^1_\varepsilon$,

$$\mathbb{P}\left(\|\theta_{t-r_\varepsilon} - \theta^\star\| \le \delta, \lim_q \theta_q = \theta^\star\right) \le \varepsilon/2 \;.$$

Hence, for any $t \geq T^1_\varepsilon$, $I_t(f) \leq \sum_{i=1}^3 I^i_t(f) + \varepsilon$, where

$$I^1_t(f) \quad = \quad \left|\mathbb{E}\left[\left(f(X_t) - P^{r_\varepsilon}_{\theta_{t-r_\varepsilon}} f(X_{t-r_\varepsilon})\right) \mathbb{1}_{\|\theta_{t-r_\varepsilon}-\theta^\star\| \leq \delta}\right]\right| \tag{10.2.57}$$

$$I^2_t(f) \quad = \quad \left|\mathbb{E}\left[\left(P^{r_\varepsilon}_{\theta_{t-r_\varepsilon}} f(X_{t-r_\varepsilon}) - \pi_{\theta_{t-r_\varepsilon}}(f)\right) \mathbb{1}_{\|\theta_{t-r_\varepsilon}-\theta^\star\| \leq \delta}\right]\right| \tag{10.2.58}$$

$$I^3_t(f) \quad = \quad \left|\mathbb{E}\left[\left(\pi_{\theta_{t-r_\varepsilon}}(f) - \pi_{\theta^\star}(f)\right) \mathbb{1}_{\|\theta_{t-r_\varepsilon}-\theta^\star\| \leq \delta}\right]\right| \, . \tag{10.2.59}$$

We first upper bound $I^1_t(f)$. For $\theta, \theta' \in \Theta$, let

$$D(\theta, \theta') = \sup_{x \in \mathbb{X}} \|P_\theta(x, \cdot) - P_{\theta'}(x, \cdot)\|_{\mathrm{TV}} \, .$$

Applying [9, Proposition 1.3.1], it comes for any $t \geq T^1_\varepsilon$,

$$\begin{aligned} I^1_t \quad &\leq \quad \mathbb{E}\left[2 \wedge \sum_{j=1}^{r_\varepsilon - 1} D(\theta_{t-r_\varepsilon+j}, \theta_{t-r_\varepsilon}) \mathbb{1}_{\|\theta_{t-r_\varepsilon}-\theta^\star\| \leq \delta}\right] \\ &\leq \quad \mathbb{E}\left[2 \wedge \sum_{j=1}^{r_\varepsilon - 1} (r_\varepsilon - j) D(\theta_{t-r_\varepsilon+j}, \theta_{t-r_\varepsilon+j-1}) \mathbb{1}_{\|\theta_{t-r_\varepsilon}-\theta^\star\| \leq \delta}\right] \, , \end{aligned}$$

where we used that for any $q, \ell > 0$ $D(\theta_{q+\ell}, \theta_q) \leq \sum_{j=1}^{\ell} D(\theta_{q+j}, \theta_{q+j-1})$. By Proposition 1, the random iteration number τ_ψ where the last projection occurs in Algorithm 10.1 is finite with probability one. Let then M_ε be such that $2\mathbb{P}(\tau_\psi \geq M_\varepsilon) \leq \varepsilon/2$, so that

$$I^1_t(f) \leq \mathbb{E}\left[2 \wedge \sum_{j=1}^{r_\varepsilon - 1} (r_\varepsilon - j) D(\theta_{t-r_\varepsilon+j}, \theta_{t-r_\varepsilon+j-1}) \mathbb{1}_{\|\theta_{t-r_\varepsilon}-\theta^\star\| \leq \delta} \mathbb{1}_{\tau_\psi \leq M_\varepsilon}\right] + \frac{\varepsilon}{2} \, .$$

Let now $T^2_\varepsilon \geq T^1_\varepsilon \vee (M_\varepsilon + r_\varepsilon)$ be such that

$$t \geq T^2_\varepsilon \Rightarrow \gamma_t \sup_{x \in \mathbb{X}, \theta \in \mathcal{W}_{M_\star+2}} \|H(x, \theta)\| \leq \delta \, .$$

Then, by recurrence and using (10.2.56), we obtain that on $\{\|\theta_{t-r_\varepsilon} - \theta_\star\| \leq \delta\}$, $\theta_{t-r_\varepsilon+j} \in \mathcal{W}_{M_\star+1}$ for all $0 \leq j \leq r_\varepsilon$. By Lemma 15 this yields for any $t \geq T^2_\varepsilon$

$$I^1_t(f) \leq C_{M_\star+1}[\sup_{x \in \mathbb{X}, \theta \in \mathcal{W}_{M_\star+2}} \|H(x, \theta)\|]^{1-2\kappa} \sum_{j=1}^{r_\varepsilon - 1} (r_\varepsilon - j) \gamma^{1-2\kappa}_{t-r_\varepsilon+j} + \frac{\varepsilon}{2} \, ,$$

and there exists $T^3_\varepsilon \geq T^2_\varepsilon$ such that $t \geq T^3_\varepsilon \Rightarrow \sup_{\{f : \|f\|_\infty \leq 1\}} I^1_t(f) \leq \varepsilon$.

We now consider $I^2_t(f)$; it holds

$$I^2_t \leq \mathbb{E}\left[\left\|P^{r_\varepsilon}_{\theta_{t-r_\varepsilon}}(X_{t-r_\varepsilon}, \cdot) - \pi_{\theta_{t-r_\varepsilon}}\right\|_{\mathrm{TV}} \mathbb{1}_{\|\theta_{t-r_\varepsilon}-\theta^\star\| \leq \delta}\right] \, .$$

By (10.2.56), $\|\theta_{t-r_\varepsilon} - \theta^\star\| \leq \delta \Rightarrow \theta_{t-r_\varepsilon} \in \mathcal{W}_{M_\star+1}$ and thus, applying Lemma 12 and (10.2.55)

$$\sup_{\{f : \|f\|_\infty \leq 1\}} I^2_t(f) \leq 2(1 - \rho_{M_\star+1})^{r_\varepsilon} \leq \varepsilon \, .$$

The derivation of the upper bound of I^3_t is similar to that of I^2_t, with Lemma 12 replaced by Lemma 14 and uses (10.2.54). Details are omitted.

10.3 Conclusion

We proved a strong law of large numbers for a stable version of AMOR, along with its ergodicity. Our algorithm adapts both its proposal and its target on the fly, which makes it a turn-key algorithm. Our results lead to a sound characterization of the target of AMOR that does not depend on the initialization of the algorithm nor on the user. This is the first theoretical analysis of an online relabeling algorithm to our knowledge, and our framework is generic enough to be applied to other online relabeling algorithms, provided that an appropriate modified acceptance ratio is used. The proof further shows how relabeling is related to vector quantization. Unlike previous work on stochastic approximation schemes for vector quantization, we make no strong assumptions on the trajectories of the process considered, rather, we ensure that the appropriate constraint is satisfied by introducing a penalization directly into the stochastic approximation framework.

We now examine possible directions for future work on the theoretical analysis of relabeling algorithms. First, following our analysis in Section 10.1, we know that after a finite (random) number of iterations, there is no projection (Steps 15 to 17 in Figure 10.1) anymore in stable AMOR, so that it should behave similarly to a penalized AMOR. The penalization, corresponding to Steps 13 and 14 in Figure 10.1, is however necessary for convergence. In most cases, since the coefficient α of the penalization is small and the limiting points are relatively far from the border of Θ, AMOR and stable AMOR should have similar behaviors, but we should investigate practical cases where AMOR converges to the border of Θ to have a better intuition of the rôle of the penalization in stable AMOR. Second, the question of the control of the convergence of AMOR arises, and results such as a central limit theorem would be a natural next step.

Chapter 11
Closing remarks

We presented in this thesis our contributions to autonomous learning and inference. Rather than summarizing them anew, we come back in this section to our vision and underlying links between chapters that we think deserve to be emphasized as closing remarks.

Racing, bandits, and SMBO

Racing algorithms, described in Section 3.2.3 are similar to pure exploration bandit algorithms [32] – bandit algorithms that seek to identify the best out of a given number of arms in a given time budget, instead of aiming at the traditional exploration/exploitation trade-off. Now, in the bandit framework, if the reward function, as a function on the arm space, can be assumed to follow a GP prior, then the problem of finding the best arm may be addressed with GP-based SMBO algorithm (see Chapter 2). In [122], for example, an SMBO approach is proposed, where the auxiliary criterion is inspired by the upper-confidence bound algorithm (UCB; [11]). The approach allows an elegant derivation of results on the speed of convergence of the algorithm, which is novel in SMBO. On the other hand, Bayesian methods for bandit problems also exist (see [83, Section 1] for a review) and provide efficient algorithms, such as the Bayes-UCB algorithm of [83]. Bayes-UCB selects the arm to play according to quantiles of the posterior on the reward function at each arm, in a manner strikingly similar to SMBO with probability of improvement auxiliary criterion [82]. We think that the bandit and the SMBO framework can be further interwoven, bringing new methods and new proof techniques to each domain.

A perpetual tuner

We foresee that hyperparameter tuning will be further automatized. Imagine a community of users that makes repeated use of a single algorithm, such as a company that specializes in the application of neural networks. Once implemented a collaborative tuning framework like SCoT (Chapter 5), the tuning process of SCoT could basically be run until the end of times on dedicated computers. When users have a new dataset, they add it to the database and it will eventually be treated. In an ideal world, imagine that all users of single-layer neural networks in the world one day agree on the choice of a hyperparameter space and dataset features, and store their datasets and tested hyperparameters on a centralized structure, and start running SCoT. Whenever somebody wants to tune a neural network on a new dataset, he logs in to the structure, uploads his dataset, and regularly receives a report specifying the best hyperparameters for his problem.

There is still methodological progress to make in order to be able to set up such a structure, but we are not that far. First, users usually have deadlines, and they will want to tune the algorithm the best way possible until the deadline is reached. Budgeted hyperparameter tuning is thus an interesting research avenue. Second, traditional GPs would probably not be the right choice for such a structure which should scale to the hopefully numerous uploaded datasets. On the one hand, substantial efforts were made to reduce the inherent cubic computational cost of training a GP, see [108, Chapter 8], and [38] for a recently available experimental comparison of the state of the art. Furthermore, other models than GPs can be plugged in SMBO, such as the bagged regression trees of [77], and we think there is room for other regression schemes.

Doubly adaptive algorithms

Although SCoT (Chapter 5) and AMOR (Chapter 9) are completely different algorithms, they share the property of being adaptive both in their proposal mechanism and in the design of their target. Furthermore, in both algorithms, the two adaptations are linked: AMOR tries to find the restriction of the target for which its Gaussian proposals are optimal, while SCoT defines its target as a smooth function that preserves the rankings it takes as input, and uses a model of this function to propose a new point to evaluate.

One could object that, unlike AMOR, SCoT does not really pick a target that makes its proposals more efficient. However, in Chapter 5, the values of the target are obtained, at each iteration, by running SVMrank, a surrogate ranking algorithm. SVMrank is based on an SVM algorithm, which is an optimization algorithm with a penalty for functions that are not smooth. More precisely, the SVM is associated to a user-defined kernel k and the penalty is high for functions with a large norm in the reproducing kernel Hilbert space corresponding to k. In Chapter 5, we used a squared exponential kernel, and thus favored targets with low frequencies, that in turn yielded a smooth expected improvement. In that sense, SCoT also allows an interplay between the adaptation of the proposal and the adaptation of the target. Now, the question arises how to match the right kernel to the right auxiliary criterion.

Adaptive MCMC and kernels

It was proposed in [93] to adapt the parameters of the proposal of an MCMC algorithm with SMBO. The approach works by setting a GP prior on the autocorrelation of the chain as a function of the parameters of the proposal distribution. Regularly, a new set of proposal parameters are chosen that maximize expected improvement of the autocorrelation. The setup is demonstrated on constrained discrete distributions, with proposals using a few hyperparameters. However, in practice, the adaptation is stopped at the end of a burn in period. We do not think this method can be easily applied to generic MCMC. Furthermore, continuing the adaptation during the entire run would certainly destroy MCMC convergence, since the behavior of the sequence of adapted parameters will be hard to control and probably erratic. Indeed, when a region of the parameter space will have been sufficiently explored, EI will be higher in unexplored regions where posterior variance of the GP is large. It can be proved that under suitable assumptions on the kernel, SMBO with EI produces dense sequences [128], which is precisely not what adaptive MCMC needs, since changes in the proposal have to progressively diminish [113, 59]. Still, we think kernels could be used advantageously in adaptive MCMC. Kernel machines [46], for instance, rely heavily on linearity in a reproducing kernel space implying nonlinear relationships in their input space, and we foresee applications of kernels to the design of better, nonlinear adaptive proposals.

Appendix A
Notations

In this appendix, we gather both notations that appear repeatedly throughout the thesis and chapter-specific remarks.

Miscellaneous

We denote the natural logarithm by log.

Probability distributions

Table A.1 summarizes the notations we used for distributions and their probability density functions (henceforth pdfs). Since Chapter 7 contains probabilistic models and products of several pdfs, it has its own notations, with distributions bearing their parameters as indices, to keep formulas clear.

Table A.1: Summary of notations for distributions and pdfs. A blank space means the object is not used.

Distribution	**Parameters**	**Notation**	**pdf**	**Notation in Chapter 7**	**pdf in Chapter 7**
1D Gaussian	mean μ, variance σ^2	$\mathcal{N}(\mu, \sigma^2)$	$\mathcal{N}(\cdot\|\mu, \sigma^2)$	$\mathcal{N}_{\mu,\sigma^2}$	$\mathcal{N}_{\mu,\sigma^2}(\cdot)$
Gaussian	mean μ, covariance matrix Σ	$\mathcal{N}(\mu, \Sigma)$	$\mathcal{N}(\cdot\|\mu, \Sigma)$		
Poisson	mean λ	$\mathrm{Poi}(\lambda)$		Poi_λ	$\mathrm{Poi}_\lambda(\cdot)$
gamma	shape k, scale θ			$\Gamma_{k,\theta}$	$\Gamma_{k,\theta}(\cdot)$
inverse gamma	shape α, scale β			$\mathcal{IG}_{\alpha,\beta}$	
Dirac	location a			δ_a	$\delta_a(\cdot)$
uniform	interval $[a, b]$	$U(a, b)$			

Notations for Part I

$\mathbb{H}$ always denotes the set of all possible hyperparameters. It is usually a product of discrete and continuous spaces. $\mathcal{O}$ denotes the set of already evaluated hyperparameters. Each point in $\mathcal{O}$ is a pair (hyperparameter, quality), where the function used as a quality measure is a validation or cross-validation error, except in Chapter 5, where a more sophisticated quality measure is defined. In Chapter 5, points in $\mathcal{O}$ become triplets (D_i, x_i, y_i), since each point is attached to a single dataset D_i.

Notations for Chapter 7

In Chapter 7, we use a large number of probability distributions with the following convention. p denotes all distributions and is thus largely overloaded: the actual semantics is defined by its arguments. If we further want to differentiate between distributions of the same or similar arguments, we put a label in the index of p. Estimates are denoted by a hat. We use (mostly small) italic letters for variables (with the notable exception of N denoting the number of muons). Vectors are typeset bold. If $\mathbf{x}$ is a vector, x_i is its ith element.

Bibliography

[1] P. Abreu et al. Update on the correlation of the highest energy cosmic rays with nearby extragalactic matter. *Astroparticle Physics*, 34:314–326, 2012. (Cited on pages 67 and 68.)

[2] B. Adenso-Diaz and M. Laguna. Fine-tuning of algorithms using fractional experimental design and local search. *Operations Research*, 54(1):99–114, 2006. (Cited on page 27.)

[3] M. Aglietta. A direct measurement of the photoelectron number per vertical muon in the Capisa SD detector. Technical Report GAP-05-010, Auger Project Technical Note, 2005. (Cited on page 74.)

[4] J. Alvarez-Muñiz, G. Rodríguez-Fernández, I. Valiño, and E. Zas. An alternative method for tank signal response and $S(1000)$ calculation. Technical Report GAP-05-054, Auger Project Technical Note, 2005. (Cited on page 74.)

[5] J. Alvarez-Muñiz, G. Rodríguez-Fernández, I. Valiño, and E. Zas. Update on the method for tank signal response (SdSignalUSC code). Technical Report GAP-09-038, Auger Project Technical Note, 2009. (Cited on page 74.)

[6] C. Andrieu, E. Moulines, and P. Priouret. Stability of stochastic approximation under verifiable conditions. *SIAM Journal on Control and Optimization*, 44:283–312, 2005. (Cited on pages 118, 121 and 147.)

[7] C. Andrieu and C. P. Robert. Controlled Markov chain Monte Carlo methods for optimal sampling. Technical Report 125, Cahiers du Ceremade, 2001. (Cited on page 102.)

[8] C. Andrieu and J. Thoms. A tutorial on adaptive MCMC. *Statistics and Computing*, 18:343–373, 2008. (Cited on pages 3 and 102.)

[9] Y. Atchadé, G. Fort, E. Moulines, and P. Priouret. *Bayesian Time Series Models*, chapter Adaptive Markov chain Monte Carlo: Theory and Methods, pages 33–53. Cambridge Univ. Press, 2011. (Cited on pages 3, 102 and 149.)

[10] C. Audet and D. Orban. Finding optimal algorithmic parameters using the mesh adaptive direct search algorithm. *SIAM Journal on Optimization*, 17(3):642 – 664, 2006. (Cited on page 27.)

[11] P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47:235–256, 2002. (Cited on pages viii and 151.)

[12] P. Auger. Extensive cosmic-ray showers. *Review of Modern Physics*, 11:288–291, 1939. (Cited on page 60.)

[13] R. Bardenet, O. Cappé, G. Fort, and B. Kégl. Adaptive MCMC with online relabeling. *Submitted, preprint available as arXiv:1210.2601.* (Cited on pages vii, 4, 101 and 117.)

[14] R. Bardenet, O. Cappé, G. Fort, and B. Kégl. An adaptive Metropolis algorithm with online relabeling. In *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 22, pages 91–99, April 2012. (Cited on pages v, vii, 3, 4, 101, 102, 103, 117, 118, 122 and 131.)

[15] R. Bardenet and B. Kégl. Surrogating the surrogate: accelerating Gaussian-process-based global optimization with a mixture cross-entropy algorithm. In *Proceedings of the 27th International Conference on Machine Learning (ICML)*, 2010. (Cited on page 9.)

[16] R. Bardenet and B. Kégl. An adaptive Monte Carlo Markov chain algorithm for inference from mixture signals. In *Proceedings of ACAT'11, Journal of Physics: Conference series*, 2012. (Cited on pages vii, 4, 76, 80, 94 and 101.)

[17] R. Bardenet, B. Kégl, and D. Veberič. Single muon response: The signal model. Technical Report GAP-10-110, Auger Project Technical Note, 2010. (Cited on pages vii, 4 and 73.)

[18] R. Benassi, J. Bect, and E. Vazquez. Bayesian optimization using sequential Monte Carlo. In *Proceedings of the International Conference on Learning and Intelligent Optimization (LION)*, 2012. (Cited on page 13.)

[19] D. Benbouzid, R. Busa-Fekete, N. Casagrande, F.-D. Collin, and B. Kégl. MultiBoost: a multi-purpose boosting package. *Journal of Machine Learning Research*, 13:549–553, 2012. (Cited on page 49.)

[20] Y. Bengio. Learning deep architectures for AI. *Foundations and Trends in Machine Learning*, 2(1):1–127, 2009. (Cited on pages iii, 1 and 31.)

[21] J. Bergstra, R. Bardenet, B. Kégl, and Y. Bengio. Algorithms for hyperparameter optimization. In *Advances in Neural Information Processing Systems*, volume 24. The MIT Press, 2011. (Cited on pages iv, v, 2, 3, 31, 38 and 52.)

[22] J. Bergstra, R. Bardenet, B. Kégl, and Y. Bengio. Implementations of algorithms for hyper-parameter optimization. In *NIPS Workshop on Bayesian optimization*, 2011. (Cited on page 29.)

[23] J. Bergstra and Y. Bengio. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 2012. (Cited on pages 35, 36 and 52.)

[24] M. Birattari, T. Stützle, L. Paquete, and K. Varrentrapp. A racing algorithm for configuring metaheuristics. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, 2002. (Cited on page 27.)

[25] C. M. Bishop. *Neural Networks for Pattern Recognition.* Oxford University Press, 1995. (Cited on pages 37 and 90.)

[26] C. B. Bonifazi, P. Bauleo, A. Ferrero, A. Filevich, and A. Reguera. Response of a water Cherenkov detector to oblique and quasi-horizontal muons. Technical Report GAP-01-018, Auger Project Technical Note, 2001. (Cited on page 74.)

[27] B. Boser, I. Guyon, and V. Vapnik. A training algorithm for optimal margin classifiers. In *Fifth Annual Workshop on Computational Learning Theory*, pages 144–152, 1992. (Cited on pages iii and 1.)

[28] S. Boyd and L. Vandenberghe. *Convex Optimization.* Cambridge University Press, 1991. (Cited on page 122.)

[29] L. Breiman. Random forests. *Machine Learning*, 2001. (Cited on page 28.)

[30] M. Brendel and M. Schoenauer. Instance-based parameter tuning for evolutionary AI planning. In *Proceedings of the 20th Genetic and Evolutionary Computation Conference*, 2011. (Cited on pages 28 and 44.)

[31] J.W. Brewer. Kronecker products and matrix calculus in system theory. *IEEE Transactions on Circuits and Systems*, 25:772–781, 1978. (Cited on pages 127 and 129.)

[32] S. Bubeck, R. Munos, and G. Stoltz. Pure exploration in multi-armed bandits problems. In *Proceedings of the international conference on Algorithmic Learning Theory (ALT)*, 2009. (Cited on pages 27 and 151.)

[33] B. P. Carlin and T. A. Louis. *Bayes and Empirical Bayes Methods for Data Analysis.* Chapman & Hall / CRC, 2000. (Cited on page 87.)

[34] M. A. Carreira-Perpiñan and G. E. Hinton. On contrastive divergence learning. In *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics (AISTATS)*, 2005. (Cited on page 35.)

[35] A. Castellina and G. Navarra. Separating the electromagnetic and muonic components in the FADC traces of the Auger Surface Detectors. Technical Report GAP-06-065, Auger Project Technical Note, 2006. (Cited on page 74.)

[36] G. Celeux. Bayesian inference for mixtures: The label-switching problem. In R. Payne and P. Green, editors, *COMPSTAT 98.* Physica-Verlag, 1998. (Cited on pages 94, 98, 102, 106 and 113.)

[37] G. Celeux, M. A. Hurn, and C.P. Robert. Computational and inferential difficulties with mixture posterior distributions. *J. American Statist. Assoc.*, 95:957–970, 1995. (Cited on pages 93, 99 and 102.)

[38] K. Chalupka, C. K. I. Williams, and I. Murray. A framework for evaluating approximation methods for Gaussian process regression. *pre-print*, 2012. arXiv:1205.6326. (Cited on pages viii, 54 and 152.)

[39] A. S. Chou. Vertical equivalent muon study with the Fermilab tank. Technical Report GAP-02-045, Auger Project Technical Note, 2002. (Cited on page 74.)

[40] W. Chu and Z. Ghahramani. Preference learning with Gaussian processes. In *Proceedings of the 22nd International Conference on Machine Learning*, pages 137–144, 2005. (Cited on pages 47 and 48.)

[41] A. Coates, H. Lee, and A. Y. Ng. An analysis of single-layer networks in unsupervised feature learning. NIPS Deep Learning and Unsupervised Feature Learning Workshop, 2010. (Cited on page iv.)

[42] A. Coates, H. Lee, and A. Y. Ng. An analysis of single-layer networks in unsupervised feature learning. NIPS Deep Learning and Unsupervised Feature Learning Workshop, 2010. (Cited on pages 1, 26 and 32.)

[43] A. Coates and A. Y. Ng. The importance of encoding versus training with sparse coding and vector quantization. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2011. (Cited on pages iii and iv.)

[44] A. Coates and A. Y. Ng. The importance of encoding versus training with sparse coding and vector quantization. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2011. (Cited on pages 1, 26 and 32.)

[45] S. P. Coy, B. L. Golden, G. C. Runger, and E. A. Wasil. Using experimental design to find effective parameter settings for heuristics. *Journal of Heuristics*, 7(1):77 – 97, 2001. (Cited on page 28.)

[46] N. Cristianini and J. Shawe-Taylor. *Kernel methods for pattern recognition.* Cambridge University Press, 2004. (Cited on pages iii, 1 and 153.)

[47] A. J. Cron and M. West. Efficient classification-based relabeling in mixture models. *The American Statistician*, pages 16–20, 2011. (Cited on page 98.)

[48] D. Ravignani et al. Calculation of the number of photoelectrons with the water Cherenkov detector model. Technical Report GAP-97-024, Auger Project Technical Note, 1997. (Cited on pages 74, 77 and 78.)

[49] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, Series B, 39(1):1–38, 1977. (Cited on pages 88 and 99.)

[50] P. N. Diep. Comments on muon counting in the FADC traces of the Auger surface detector. Technical Report GAP-08-136, Auger Project Technical Note, 2008. (Cited on page 74.)

[51] D. Dornic. *Développement et caractérisation de photomultiplicateurs hémisphériques pour les expériences d'astroparticules d'étalonnage des détecteurs de surface et analyse des gerbes horizontales de l'Observatoire Pierre Auger*. PhD thesis, Université Paris XI, 2006. pdf. (Cited on pages 74, 76, 77 and 78.)

[52] D. Dornic, F. Arneodo, I. Lhenry-Yvon, X. Bertou, C. Bonifazi, P. Ghia, C. Grunfeld, and T. Suomijarvi. Calibration analysis: Capisa data. Technical Report GAP-05-101, Auger Project Technical Note, 2005. (Cited on page 74.)

[53] A. Etchegoyen. Track geometry and smearing of the bump calibration. Technical Report GAP-02-078, Auger Project Technical Note, 2002. (Cited on page 74.)

[54] FCEyN & Tandar Groups. Simulations with GEANT. Technical Report GAP-96-011, Auger Project Technical Note, 1996. (Cited on page 74.)

[55] G. R. Fernandez, A. Tripathi, and K. Arisaka. Simulation of the Pierre Auger surface detector response using GEANT4. Technical Report GAP-03-054, Auger Project Technical Note, 2003. (Cited on page 74.)

[56] G. R. Fernandez, E. Zas, T. Ohnuki, A. Tripathi, D. Barnhill, J. Lee, K. Arisaka, and W. E. Slater. Surface detector response using lookup table based on GEANT4 simulation. Technical Report GAP-04-045, Auger Project Technical Note, 2004. (Cited on page 74.)

[57] D. Garcia-Pinto for the Pierre Auger collaboration. Measurements of the longitudinal development or air showers with the Pierre Auger observatory. In *Proceedings of the International Cosmic Ray Conference*, 2012. (Cited on pages 70 and 72.)

[58] F. Salamida for the Pierre Auger collaboration. Update on the measurement of the CR energy spectrum above 10^{18} eV made using the Pierre Auger observatory. In *Proceedings of the International Cosmic Ray Conference*, 2012. (Cited on pages 67 and 69.)

[59] G. Fort, E. Moulines, and P. Priouret. Convergence of adaptive and interacting Markov chain Monte Carlo algorithms. *Annals of Statistics*, 39(6):3262–3289, 2012. (Cited on pages 118, 143, 147, 148 and 153.)

[60] S. Frühwirth-Schnatter. Markov chain Monte Carlo estimation of classical and dynamic switching in mixture models. *Journal of the American Statistical Association*, 96(453):194–209, 2001. (Cited on page 96.)

[61] X. Garrido, A. Cordier, S. Dagoret-Campagne, B. Kégl, D. Monnier-Ragaigne, and M. Urban. Measurement of the number of muons in Auger tanks by the FADC jump counting method. Technical Report GAP-07-060, Auger Project Technical Note, 2007. (Cited on page 74.)

[62] X. Garrido, B. Kégl, A. Cordier, S. Dagoret-Campagne, D. Monnier-Ragaigne, and M. Urban. Update and new results from the FADC jump counting method. Technical Report GAP-09-023, Auger Project Technical Note, 2009. (Cited on page 74.)

[63] B. Genolini, T. Nguyen Trunc, J. Pouthas, P. Lavoute, C. Meunier, M. Aglietta, and C. Morello. Photonis XP1805 and PAO SD bases: effects of the temperature and of the Earth's magnetic field. Technical Report GAP-03-017, Auger Project Technical Note, 2003. (Cited on pages 77 and 78.)

[64] P. W. Goldberg, C. K. I. Williams, and C. M. Bishop. Regression with input-dependent noise: A Gaussian process treatment. In *Advances in Neural Information (NIPS)*, 1998. (Cited on page 13.)

[65] S. Graf and H. Luschgy. *Foundations of Quantization for Probability Distributions*. Springer-Verlag, 2000. (Cited on pages 103, 122 and 124.)

[66] R. Gramacy. *Bayesian treed Gaussian process models*. PhD thesis, UC Santa Cruz, 2005. (Cited on page 30.)

[67] J. Gratch and G. Dejong. Composer: A probabilistic solution to the utility problem in speed-up learning. In P. Rosenbloom and P. Szolovits, editors, *Proceedings of the Tenth National Conference on Artificial Intelligence (AAAI)*, pages 235 – 240, 1992. (Cited on page 26.)

[68] P. J. Green. Reversible jump Markov chain Monte Carlo computation and Bayesian model determination. *Biometrika*, 82(4):711–732, 1995. (Cited on pages 23, 81 and 115.)

[69] H. Haario, E. Saksman, and J. Tamminen. An adaptive Metropolis algorithm. *Bernoulli*, 7:223–242, 2001. (Cited on pages vi, 3, 23, 94, 102 and 104.)

[70] P. Hall and C. C. Heyde. *Martingale limit theory and its application*. Academic Press, New York, 1980. (Cited on page 144.)

[71] N. Hansen. The CMA evolution strategy: a comparing review. In J.A. Lozano, P. Larranaga, I. Inza, and E. Bengoetxea, editors, *Towards a new evolutionary computation. Advances on estimation of distribution algorithms*, pages 75–102. Springer, 2006. (Cited on pages 22, 28, 33 and 53.)

[72] P. Hennig and C. J. Schuler. Entropy search for information-efficient global optimization. *Journal of Machine Learning Research*, 2012. (Cited on page 10.)

[73] V. Hess. Observations of the penetrating radiation on seven balloon flights. *Physik. Zeitschr.*, 13:1084–1091, 1912. (Cited on page 60.)

[74] G. E. Hinton, S. Osindero, and Y. Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18:1527–1554, 2006. (Cited on page 31.)

[75] S. Hornus and J. Boissonnat. An efficient implementation of delaunay triangulations in medium dimensions. Technical Report RR-6743, INRIA, 2008. (Cited on page 18.)

[76] M. A. Hurn, A. Justel, and C.P. Robert. Estimating mixtures of regressions. *Journal of Computational and Graphical Statistics*, 12:55–79, 2003. (Cited on page 99.)

[77] F. Hutter. *Automated Configuration of Algorithms for Solving Hard Computational Problems.* PhD thesis, University of British Columbia, 2009. (Cited on pages viii, 25, 28, 29 and 152.)

[78] A. Hyvärinen and E. Oja. Independent component analysis: Algorithms and applications. *Neural Networks*, 13(4–5):411–430, 2000. (Cited on page 35.)

[79] A. Jasra. *Bayesian inference for mixture models via Monte Carlo.* PhD thesis, Imperial College London, 2005. (Cited on pages 94, 98, 102 and 115.)

[80] A. Jasra, C. C. Holmes, and D. A. Stephens. Markov chain Monte Carlo methods and the label switching problem in Bayesian mixture modelling. *Statistical Science*, 20(1):50–67, 2005. (Cited on pages 93, 94 and 102.)

[81] T. Joachims. Optimizing search engines using clickthrough data. In *Proceedings of the ACM Conference on Knowledge Discovery and Data Mining (KDD)*, 2002. (Cited on pages 47 and 48.)

[82] D. R. Jones. A taxonomy of global optimization methods based on response surfaces. *Journal of Global Optimization*, 21:345–383, 2001. (Cited on pages iv, v, 2, 10, 12, 44 and 151.)

[83] E. Kaufmann, O. Cappé, and A. Garivier. On Bayesian upper confidence bounds for bandit problems. In *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2012. (Cited on pages viii, 30 and 151.)

[84] B. Kégl and R. Busa-Fekete. Boosting products of base classifiers. In *International Conference on Machine Learning*, volume 26, pages 497–504, Montreal, Canada, 2009. (Cited on page 49.)

[85] B. Kégl, R. Busa-Fekete, K. Louedec, R. Bardenet, X. Garrido, I.C. Mariş, D. Monnier-Ragaigne, S. Dagoret-Campagne, and M. Urban. Reconstructing $N_{\mu 19}(1000)$. Technical Report GAP-11-054, Auger Project Technical Note, 2011. (Cited on pages 73, 74, 83, 84, 89 and 90.)

[86] B. Kégl, M. Unger, and R. Busa-Fekete. A nonparametric approach to estimate $X_{\max}$ using Gaussian process regression. Technical Report GAP-10-034, Auger Project Technical Note, 2010. (Cited on page 87.)

[87] B. Kégl and D. Veberič. Single muon response: Tracklength. Technical Report GAP-09-043, Auger Project Technical Note, 2009. (Cited on pages 75 and 76.)

[88] H. Larochelle, D. Erhan, A. Courville, J. Bergstra, and Y. Bengio. An empirical evaluation of deep architectures on problems with many factors of variation. In *Proceedings of the International Conference on Machine Learning*, pages 473–480, 2007. (Cited on pages 32, 35, 36, 38 and 41.)

[89] P. Larrañaga and J. Lozano, editors. *Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation*. Springer, 2001. (Cited on pages 16, 18, 19 and 27.)

[90] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, November 1998. (Cited on page 31.)

[91] D. Lizotte. *Practical Bayesian Optimization*. PhD thesis, University of Alberta, 2008. (Cited on page 44.)

[92] K. Louedec. *Atmospheric aerosols at the Pierre Auger Observatory: characterization and effect on the energy estimation for ultra-high energy cosmic rays*. PhD thesis, Université Paris-Sud XI, 2011. (Cited on page 59.)

[93] N. Mahendran, Z. Wang, F. Hamze, and N. de Freitas. Adaptive MCMC with Bayesian optimization. In *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2012. (Cited on page 152.)

[94] J. M. Marin, K. Mengersen, and C.P. Robert. Bayesian modelling and inference on mixtures of distributions. *Handbook of Statistics*, 25, 2004. (Cited on pages 94, 96, 102 and 106.)

[95] A. McHutchon and C. E. Rasmussen. Gaussian process training with input noise. In *Advances in Neural Information (NIPS)*, 2011. (Cited on page 13.)

[96] S. P. Meyn and R.L. Tweedie. *Markov chains and stochastic stability*. Springer, 1993. (Cited on page 135.)

[97] J. Mockus, V. Tiesis, and A. Zilinskas. The application of Bayesian methods for seeking the extremum. In L.C.W. Dixon and G.P. Szego, editors, *Towards Global Optimization*, volume 2, pages 117–129. North Holland, New York, 1978. (Cited on page 10.)

[98] M. Muja and D. G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In *International Conference on Computer Vision Theory and Applications (VISAPP)*, 2009. (Cited on page 27.)

[99] V. Nannen, S. Smit, and Eiben A. Costs and benefits of tuning parameters of evolutionary algorithms. In *Parallel Problem Solving from Nature (PPSN)*, 2008. (Cited on page 27.)

[100] G. Pagès. A space quantization method for numerical integration. *Journal of Computational and Applied Mathematics*, 89:1–38, 1997. (Cited on pages 103 and 123.)

[101] P. Papastamoulis and G. Iliopoulos. An artificial allocations based solution to the label switching problem in Bayesian analysis of mixtures of distribution. *Journal of Computational and Graphical Statistics*, 19:313–331, 2010. (Cited on pages 94, 97 and 102.)

[102] P. Papastamoulis and G. Iliopoulos. On the convergence rate of random permutation sampler and ECR algorithm in missing data models. *Methodology and Computing in Applied Probability*, 2011. (Cited on page 97.)

[103] N. Pinto, D. Doukhan, J. J. DiCarlo, and D. D. Cox. A high-throughput screening approach to discovering good forms of biologically inspired visual representation. *PLoS Comput Biol*, 5(11):e1000579, 11 2009. (Cited on pages iii, iv, 1, 26, 32 and 35.)

[104] F. P. Preparata and M. I. Shamos. *Computational Geometry, an Introduction*. Texts and Monographs in Computer Science. Springer-Verlag, 1988. (Cited on page 18.)

[105] C. Pryke. Geometrical design studies for water Cherenkov detectors via simulation. Technical Report GAP-96-008, Auger Project Technical Note, 1996. (Cited on page 74.)

[106] C. Pryke. Performance simulations of a 10 m^2 water Cherenkov detector and comparison with experiment. Technical Report GAP-97-004, Auger Project Technical Note, 1997. (Cited on page 74.)

[107] C. Pryke. Self calibration of the water Cherenkov tanks: Simulation. Technical Report GAP-97-026, Auger Project Technical Note, 1997. (Cited on page 74.)

[108] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006. (Cited on pages v, viii, 12, 13, 21, 28, 37, 47, 48, 54, 87 and 152.)

[109] S. Richardson and P. J. Green. On Bayesian analysis of mixtures with an unknown number of components. *Journal of the Royal Statistical Society, Series B*, 59(4):731–792, 1997. (Cited on page 102.)

[110] C. P. Robert and G. Casella. *Monte Carlo Statistical Methods*. Springer-Verlag, New York, 2004. (Cited on pages iv, 2, 15, 17 and 74.)

[111] G. Roberts, A. Gelman, and W. Gilks. Weak convergence of optimal scaling of random walk Metropolis algorithms. *The Annals of Applied Probability*, 7:110–120, 1997. (Cited on pages vi, 3, 102 and 104.)

[112] G. O. Roberts and J. S. Rosenthal. Optimal scaling for various Metropolis-Hastings algorithms. *Statistical Science*, 16:351–367, 2001. (Cited on pages vi, 3, 102 and 104.)

[113] G. O. Roberts and J. S. Rosenthal. Coupling and ergodicity of adaptive MCMC. *Journal of Applied Probability*, 44:486–475, 2007. (Cited on pages 118 and 153.)

[114] G. O. Roberts and J. S. Rosenthal. Examples of adaptive MCMC. *Journal of Computational and Graphical Statistics*, 18:349–367, 2009. (Cited on page 102.)

[115] A. Roodaki. *Signal decompositions using trans-dimensional Bayesian methods.* PhD thesis, Supélec, 2012. (Cited on page 94.)

[116] A. Roodaki, J. Bect, and G. Fleury. Summarizing posterior distributions in signal decomposition problems when the number of components is unknown. In *IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, Kyoto, Japan, 2012. (Cited on pages 94 and 102.)

[117] R. Y. Rubinstein and D. P. Kroese. *The Cross-Entropy Method: A Unified Approach to Combinatorial Optimization, Monte Carlo Simulation and Machine Learning*. Springer, 2004. (Cited on pages 15, 16 and 19.)

[118] R. E. Schapire and Y. Singer. Improved boosting algorithms using confidence-rated predictions. *Machine Learning*, 37(3):297–336, 1999. (Cited on page 45.)

[119] W. E. Slater, A. Tripathi, and K. Arisaka. A GEANT3 simulation of Pierre Auger Surface Detector response to muons. Technical Report GAP-02-063, Auger Project Technical Note, 2002. (Cited on page 74.)

[120] J. Snoek, H. Larochelle, and R. P. Adams. Practical Bayesian optimization of machine learning algorithms. In *Advances in Neural Information Processing Systems (NIPS)*, 2012. (Cited on pages iv, 2, 10, 13, 29 and 38.)

[121] M. Sperrin, T. Jaki, and E. Wit. Probabilistic relabelling strategies for the label switching problem in Bayesian mixture models. *Statistics and Computing*, 20:357–366, 2010. (Cited on pages 94, 99, 102 and 115.)

[122] N. Srinivas, A. Krause, S. Kakade, and M. Seeger. Gaussian process optimization in the bandit setting: No regret and experimental design. In *Proceedings of the 27th International Conference on Machine Learning*, 2010. (Cited on pages viii, 10, 12, 30 and 151.)

[123] M. Stephens. Dealing with label switching in mixture models. *Journal of the Royal Statistical Society, Series B*, 62:795–809, 2000. (Cited on pages 93, 94, 97, 98, 100, 102 and 110.)

[124] D. Supanitsky and X. Bertou. Semi-analytical model of the three-fold charge spectrum in a water Cherenkov tank. Technical Report GAP-03-113, Auger Project Technical Note, 2003. (Cited on page 74.)

[125] H. Terashima-Marín, P. Ross, and M. Valenzuela-Réndon. Evolution of constraint satisfaction strategies in examination timetabling. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, pages 635 – 642, 1999. (Cited on page 27.)

[126] C. Thornton, F. Hutter, H. H. Hoos, and K. Leyton-Brown. Auto-weka: Automated selection and hyper-parameter optimization of classification algorithms. Technical Report TR-2012-05, University of British Columbia, Department of Computer Science, 2012. (Cited on pages iv, 2 and 29.)

[127] B. A. Tolson and C. A. Shoemaker. Dynamically dimensioned search algorithm for computationally efficient watershed model calibration. *Water Resources Research*, 43, 2007. (Cited on page 27.)

[128] E. Vazquez and J. Bect. Convergence properties of the expected improvement algorithm with fixed mean and covariance functions. *Journal of Statistical Planning and Inference*, 140(11):3088–3095, 2010. (Cited on page 153.)

[129] J. Villemonteix, E. Vazquez, and E. Walter. An informational approach to the global optimization of expensive-to-evaluate functions. *Journal of Global Optimization*, 2006. (Cited on pages 10, 12 and 14.)

[130] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P. A. Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Machine Learning Research*, 11:3371–3408, 2010. (Cited on page 31.)

Abstract

Inference and optimization algorithms usually have hyperparameters that require to be tuned in order to achieve efficiency. We consider here different approaches to efficiently automatize the hyperparameter tuning step by learning online the structure of the addressed problem. The first half of this thesis is devoted to the problem of hyperparameter tuning in machine learning, where recent results suggest that with current generation hardware, the optimal allocation of computing time includes more hyperparameter exploration than has been typical in the literature. After presenting and improving the generic sequential model-based optimization (SMBO) framework, we show that SMBO successfully applies to the challenging task of tuning the numerous hyperparameters of deep belief networks, outperforming expert manual tuning. To close the first part, we propose an algorithm that performs tuning *across* datasets, further closing the gap between automatized tuners and human experts by mimicking the memory that humans have of past experiments with the same algorithm on different datasets. The second half of this thesis deals with adaptive Markov chain Monte Carlo (MCMC) algorithms, sampling-based algorithms that explore complex probability distributions while self-tuning their internal parameters on the fly. This second part starts by describing the Pierre Auger observatory (henceforth Auger), a large-scale particle physics experiment dedicated to the observation of atmospheric showers triggered by cosmic rays. These showers are wide cascades of elementary particles raining on the surface of Earth, resulting from charged nuclei hitting our atmosphere with the highest energies ever seen. The analysis of Auger data motivated our study of adaptive MCMC, since the latter can cope with the complex and high-dimensional generative models involved in Auger. We derive the first part of the Auger generative model and introduce a procedure to perform inference on shower parameters that requires only this bottom part. Our generative model inherently suffers from permutation invariance, thus leading to *label switching*. Label-switching is a common difficulty in MCMC inference which makes marginal inference useless because of redundant modes of the target distribution. After reviewing previously existing solutions to the label switching problem, we propose AMOR, the first adaptive MCMC algorithm with online relabeling. We empirically demonstrate the benefits of adaptivity and show how AMOR satisfyingly applies to the problem of inference in our Auger model. Finally, we prove consistency results for a variant of AMOR. Our proof provides a generic framework for the analysis of other relabeling algorithms and unveils interesting links between relabeling algorithms and vector quantization.