The Cyclohedron Test for Finding Periodic Genes in Time Course Expression Studies

Jason Morton∗ Lior Pachter†
Anne Shiu‡ Bernd Sturmfels**

∗University of California, Berkeley, morton@math.psu.edu
†University of California, Berkeley, lpachter@math.berkeley.edu
‡University of California, Berkeley, annejls@math.berkeley.edu
**University of California, Berkeley, bernd@math.berkeley.edu

Copyright ©2007 The Berkeley Electronic Press. All rights reserved.
The Cyclohedron Test for Finding Periodic Genes in Time Course Expression Studies

Jason Morton, Lior Pachter, Anne Shiu, and Bernd Sturmfels

Abstract

The problem of finding periodically expressed genes from time course microarray experiments is at the center of numerous efforts to identify the molecular components of biological clocks. We present a new approach to this problem based on the cyclohedron test, which is a rank test inspired by recent advances in algebraic combinatorics. The test has the advantage of being robust to measurement errors, and can be used to ascertain the significance of top-ranked genes. We apply the test to recently published measurements of gene expression during mouse somitogenesis and find 32 genes that collectively are significant. Among these are previously identified periodic genes involved in the Notch/FGF and Wnt signaling pathways, as well as novel candidate genes that may play a role in regulating the segmentation clock. These results confirm that there are an abundance of exceptionally periodic genes expressed during somitogenesis. The emphasis of this paper is on the statistics and combinatorics that underlie the cyclohedron test and its implementation within a multiple testing framework.

KEYWORDS: time course microarray, biological clock, somitogenesis, multiple testing
1 Introduction

The search for the molecular components of biological clocks is an important first step towards understanding the regulatory mechanisms underlying periodic behavior at the molecular level. Examples of clocks that have been studied include the circadian clock [McDonald and Rosbash (2001)], the respiratory cycle clock in yeast [Klevecz et al. (2004), Spellman et al. (1998)] and the segmentation clock in vertebrates [Pourquié (2003)]. In order to find clock-related genes in a high-throughput fashion, time course array experiments are performed to measure the expression levels of genes on a genome-wide scale. This is followed by a statistical analysis to find periodically expressed genes. The analysis is non-trivial for reasons that include noisy measurements, variable times between experiments, vague notions of periodicity, and loss of power due to multiple testing.

The question of how best to analyze cyclic time series is a topic of extensive research in statistics; see [Chatfield (1978), Lu et al. (2004)]. Recent approaches, proposed in the context of microarray analysis include splines and other curve approximations [Luan and Li (2004), Storey et al. (2005)] methods based on signal processing techniques such as the Lomb-Scargle test [Glynn et al. (2006)], and non-parametric rank tests [Willbrand et al. (2005)]. All of these methods address, to varying degrees, the difficulties outlined above, and are sometimes developed in response to the specific needs dictated by individual experiments.

In this paper we introduce a new test for finding periodic genes. Our method belongs to the family of convex rank tests in [Morton et al. (2007), Section 5]. These tests were inspired by up-down analysis, the method of [Willbrand et al. (2005)]. They are based on recent advances in algebraic combinatorics, namely the theory of graph associahedra [Fomin and Reading (2004), Hohlweg and Lange (2005), Markl (1999)]. The connection between rank tests and polytopes was first suggested in [Cook and Seiford (1983)]. When using rank tests, an expression time-course is represented by a permutation. This has the advantage of providing robustness to noise, monotonic transformations, and uncertainty with respect to the underlying probability distributions, and the disadvantage of precluding a parametric analysis of the untransformed time courses. In up-down analysis, each permutation of \{1,2,\ldots,n\} is mapped to a sign vector, or signature, that records, for each adjacent pair on the \(n\)-path, which of the two measurements is higher. Significance is determined by counting the number of permutations that have an observed signature.

Our cyclohedron test is based on a similar permutation count to that of up-
down analysis, but the data points are now compared at longer range along the edges of the $n$-cycle. The cyclohedron $C_n$ is the graph associahedron when the graph $G$ is the $n$-cycle, and the cyclohedron test is the greedy method for linear programming on $C_n$. It is equivalent to the test denoted by $\tau^*_K(G)$ in [Morton et al. (2007), Section 5]. Cyclohedra are also known as Bott-Taubes polytopes, and they play an important role in representation theory [Fomin and Reading (2004), Section 3.2], combinatorics [Hohlweg and Lange (2005), Sandman (2004)], and homotopy theory [Markl (1999)]. Connections to statistical learning theory were explored and developed in [Hemmecke et al. (2006), Morton et al. (2007)].

The cyclohedron test is explained in detail in Section 2. Our presentation is elementary and self-contained. In Section 3 we present a method for assigning $p$-values to top-ranked groups of genes. This is done within a multiple hypothesis testing framework, which is compatible with any rank test for permutation data, including up-down analysis. In Sections 5 and 6 we develop the combinatorial details and efficient algorithms for the cyclohedron test. Our R code is available online, and its use is described in the Appendix.

We apply the cyclohedron test to data reported in [Dequéant et al. (2006)], consisting of 17 distinct expression array experiments from the presomitic mesoderne tissue of mouse embryos. These data were chosen because of the analyses already undertaken and the possibility for biological validation. Results are discussed in Section 4. We find that although the high-throughput array experiments are effective for finding groups of genes likely to be involved with clock regulation, multiple testing issues preclude the assignment of significance to any individual gene on the basis of periodic-looking patterns alone.

## 2 The cyclohedron test

The cyclohedron test is appropriate when seeking to determine whether a time course expression is periodic. Within a single hypothesis setting, the null hypothesis states that a gene or other unit of interest does not exhibit cyclic expression. The cyclohedron test provides a test statistic, which we call the permutation count, that replaces this vague null hypothesis. The test applies to data vectors $v = (v_1, \ldots, v_n)$ whose coordinates are distinct real numbers. The coordinates $v_i$ are measurements of the same quantity at distinct points. In our applications, the ordering of each vector should be with respect to some ‘cyclic’ time, so that any $v' = (v_i, v_{i+1}, \ldots, v_n, v_1, \ldots, v_{i-1})$ is an equally meaningful ordering. For example, the data vectors $v$ we analyze in Section 4 are ordered within a somite-formation cycle; so $v_j$ is a measurement taken
before \(v_{j+1}\) in the cycle, where \(j + 1\) is understood \(\mod n\).

The following procedure computes, for any given data vector \(v\), its signature \(\sigma(v)\) and its permutation count \(c(v)\). The signature is an unordered set \(\sigma = \{\sigma_1, \sigma_2, \ldots, \sigma_{n-1}\}\) of subsets of \(\{1, 2, \ldots, n\}\) and the permutation count is a positive integer.

**Algorithm 2.1.** (Cyclohedron test)

**Input:** A vector \(v = (v_1, \ldots, v_n)\) of distinct real numbers.

**Output:** The signature \(\sigma = \{\sigma_1, \sigma_2, \ldots, \sigma_{n-1}\}\) and permutation count \(c\).

Initialize \(c := 1\).

For \(i\) from 1 to \(n-1\), do

1. Initialize \(\sigma_i = \emptyset\), the empty set.
2. Let \(\delta_i\) be the index such that \(v_{\delta_i}\) is the \(i\)-th largest coordinate of \(v\).
3. Initialize \(\text{Left} := \emptyset\) and \(\text{Right} := \emptyset\).
4. For \(k\) from 1 to \(i-1\), do
   
   - if \(\sigma_k\) contains \(\delta_i - 1\) (modulo \(n\)) then set \(\text{Left} := \sigma_k\),
   - if \(\sigma_k\) contains \(\delta_i + 1\) (modulo \(n\)) then set \(\text{Right} := \sigma_k\).
5. Set \(\sigma_i := \{\delta_i\} \cup \text{Right} \cup \text{Left}\) and \(c := c \cdot \left(\frac{|\text{Right}| + |\text{Left}|}{|\text{Right}|}\right)\).

Let \(C_n\) denote the set of all signatures \(\sigma(v)\) as \(v\) runs over \(\mathbb{R}^n\). Algorithm 2.1 constructs not only the signature \(\sigma\) and the permutation count \(c\) but also the descent order permutation \(\delta = (\delta_1, \delta_2, \ldots, \delta_n)\) of the data vector \(v = (v_1, v_2, \ldots, v_n)\). Since \(\sigma(v)\) depends only on the descent order permutation \(\delta\), our algorithm specifies a map \(\delta \mapsto \sigma\) from the symmetric group \(\Sigma_n\) onto the set \(C_n\). For \(n \geq 4\), this map is not injective, and we are interested in the cardinalities of the preimages. For instance, the permutations \(\delta = (1, 3, 2, 4)\) and \(\delta' = (3, 1, 2, 4)\) have the same signature \(\sigma(\delta) = \sigma(\delta') = \{(\{1\}, \{3\}, \{1, 2, 3\}\}\).

The test statistic, the permutation count \(c\), is the number of permutations having the same signature as the permutation of interest. Significant data vectors have small test statistics, because it is unlikely that a random permutation will have a topographical map shared by few permutations. The permutation count \(c = c(v)\) has the following interpretation. Suppose that an appropriate null data generating distribution for each data vector \(v\) induces the uniform distribution on all descent order permutations \(\delta\) in the symmetric group \(\Sigma_n\). Note that this assumption is valid if the coordinates of the data vector are independent and identically distributed under the null distribution, so our test is therefore broadly applicable. For each signature \(\sigma \in C_n\), let \(p(\sigma)\) denote the probability that the signature \(\sigma\) would be observed under such a null distribution. The following proposition states that \(p(\sigma)\) is the fraction of permutations \(\delta\) that map to \(\sigma\).
Proposition 2.2. The permutation count $c$ computed by Algorithm 2.1 depends only on the signature $\sigma$. It equals the number of permutations $\delta$ that are mapped to $\sigma$, and hence

$$c = c(\sigma) = p(\sigma) \cdot n!.$$ 

Proof. For each $\sigma_i$ in the signature $\sigma$, at most two other sets $\sigma_j$ and $\sigma_k$ are contained in $\sigma_i$ and are maximal with this property. Here $\sigma_j$ and $\sigma_k$ are necessarily disjoint. The permutation count $c$ is the product of the corresponding binomial coefficients $\binom{|\sigma_j \cup \sigma_k|}{|\sigma_j|}$. It depends only on $\sigma$. The second statement is proved by induction on $n$, using the fact that any valid permutation of $\sigma_j$ can be shuffled with any valid permutation of $\sigma_k$, and augmented by $\delta_i$, to get a valid permutation for $\sigma_i$. Carrying out this process until $i = n$, with $\sigma_n = \{1, 2, \ldots, n\}$, yields precisely all permutations $\delta$ with signature $\sigma$. \qed

The other output of the algorithm, the signature $\sigma(v)$, can be viewed as a topographic map on the $n$-cycle that captures the shape of the data $v$. Algorithm 2.1 is an iterative procedure for drawing this topographic map. Namely, we encircle the vertices of the $n$-cycle in decreasing order of their corresponding data vector coordinates, that is, in the order $\delta_1, \delta_2, \ldots, \delta_{n-1}$. (The first circle is the set $\sigma_1$, the second is $\sigma_2$, and so on.) We do this according to the following provision: in order to encircle $\delta_i$, if it is adjacent to some vertex $j$ which has already been encircled by some $\sigma_k$, then $\sigma_i$ must contain the $\sigma_k$ circle. Accordingly, the sets “Left” and “Right” keep track of how far to the left and right $\sigma_i$ must extend. The result is an unordered set $\sigma$ of $n-1$ encircled sets $\sigma_1, \sigma_2, \ldots, \sigma_{n-1}$. Figure 1 displays the beginning of an example of this encircling process for $n = 11$.

We say that the height $h_i$ of the $i$-th vertex in the topographic map for $v$ is the number of sets $\sigma_j$ which contain $i$. We can identify the signature $\sigma$ with the height vector $h = (h_1, h_2, \ldots, h_n)$, because $\sigma$ can be recovered uniquely from the vector $h$. The map $v \mapsto h(v)$ can be viewed as a smoothing of the data; see Figure 2.

Remark 2.3. The cyclohedron test applies when there are no ties $v_i = v_j$ in the data. When ties occur, we examine all possible permutations $\delta$ arising from small perturbations.

Example 2.4. In our analysis in Section 4, the number of microarray experiments is $n = 17$, and the number of probesets (labels of the data vectors) is $N = 13,873$. The probeset ranked first in Table 1 is for a gene named Obox.
Figure 1: Algorithm 2.1 constructs a topographic map on the n-cycle by subsequently encircling vertices in order of decreasing size of the corresponding component of a data vector. Displayed at the top are the formations of the first two components $\sigma_i$, and at the bottom are the third and fourth, of the signature for an example with $n = 11$.

Its data vector equals

$$v = (0.738, 0.996, 0.705, 0.150, -0.566, -0.673, 0.774, -0.736, -0.788, -0.802, -1.276, -0.521, 0.238, -0.258, -0.249, -0.084, -0.117).$$

The descent order permutation for this vector $v$ equals

$$\delta = (2, 7, 1, 3, 13, 4, 16, 17, 15, 14, 12, 5, 6, 8, 9, 10, 11).$$

The signature $\sigma$ is given by the unordered set $\sigma_1 = \{2\}$, $\sigma_2 = \{7\}$, $\sigma_3 = \{1, 2\}$, $\sigma_4 = \{1, 2, 3\}$, etc. The permutation count $c = 480$ is the product of the three contributions made by 5, 8 and 12, respectively, when constructing $\sigma_8 = \{1, 2, 3, 4, 16, 17\}$, $\sigma_{10} = \{1, 2, 3, 4, 13, 14, 15, 16, 17\}$, and $\sigma_{13} = \{1, 2, 3, 4, 5, 6, 7, 12, 13, 14, 15, 16, 17\}$. When viewing $\sigma$ as a topographic map
for the data \( v \), we obtain the height vector
\[
h(v) = (12, 13, 11, 10, 5, 4, 5, 3, 2, 1, 0, 6, 8, 7, 8, 10, \ldots)
\]

Figure 2 displays both the data vector \( v \) and the height vector \( h(v) \) plotted around the circle. □

![Figure 2: The data v (left) and the height vector h(v) (right) of the gene 0box.](image)

### 3 Significance testing

Multiple hypothesis testing is of concern in microarray experiments, because the number of hypotheses that are tested simultaneously is large. In our application, there are \( N = 13,873 \) null hypotheses. The hypotheses take the form “the \( r \) genes with the smallest counts \( c \) arose by chance”, for \( r = 1, 2, \ldots, N \). In this section, we explain how to assign p-values to these groups, leading to a criterion for determining which hypotheses to reject.

Applying the cyclohedron test to \( N \) data vectors \( v^{(1)}, \ldots, v^{(N)} \) in \( \mathbb{R}^n \) means computing their permutation counts \( c(v^{(1)}), \ldots, c(v^{(N)}) \). The highest ranked data are those for which \( c(v^{(i)}) \) is smallest. Under the null hypothesis, the probability distribution on \( \mathbb{R}^n \) of each data vector \( v^{(i)} \) induces the uniform distribution \( U \) on the \( n! \) permutations \( \delta \). Viewed as a random variable, the permutation count \( c \) has probability distribution function
\[
P_c : \text{im}(c) \rightarrow [0, 1], \quad \gamma \mapsto \Pr_U(c(\delta) = \gamma).
\]

Here \( \text{im}(c) = \{\gamma_1 < \gamma_2 < \cdots < \gamma_{s_n}\} \) is the set of all positive integers that arise as permutation counts \( c(\sigma) \) for some \( \sigma \in C_n \). The probability distribution
function $P_c$ is displayed in Figure 3 for $n = 17$, which will be the number of time points in Section 4.

Figure 3: The probability distribution function of $c$ for $n = 17$.

We now fix two integers $1 \leq r \leq N$. The order statistic $C_r$ is the function $\mathbb{R}^{N \times n} \rightarrow \text{im}(c)$ which takes any list of $N$ data vectors $V = (v^{(1)}, \ldots, v^{(N)})$ and returns the $r$-th smallest value among the permutation counts $c(v^{(1)}), \ldots, c(v^{(N)})$. Recall that under the null hypothesis, each $v^{(i)}$ has a distribution on $\mathbb{R}^n$ which induces the uniform distribution on permutations $\delta$. Further let us assume that the data vectors $v^{(i)}$ are independent. This induces a joint distribution $Q_0$ on the vector $(c(v^{(1)}), \ldots, c(v^{(N)})) \in \mathbb{R}^{N \times n}$ of counts. In this framework, we view the order statistic $C_r$ as a random variable with distribution

$$F_r(c) : \text{im}(c) \rightarrow [0,1], \gamma \mapsto \Pr_{Q_0}(C_r = \gamma).$$

In other words, $F_r(\gamma)$ is the probability that the $r$-th smallest value among the permutation counts $c(v^{(1)}), \ldots, c(v^{(N)})$ of $N$ random data vectors equals
\(\gamma\). The function \(F_r(\gamma)\) depends only on \(n, N\) and \(r\). Its efficient computation is explained in Section 6.

**Definition 3.1. (p-value)** Suppose we apply the cyclohedron test to \(N\) data vectors in \(\mathbb{R}^n\), and the data vector whose permutation count is the \(r\)-th smallest has permutation count \(\gamma_k\). Then the collective \(p\)-value of the group of \(r\) highest ranked data vectors is

\[
\Pr_{Q_0}(C_r \leq \gamma_k) = F_r(\gamma_1) + F_r(\gamma_2) + \cdots + F_r(\gamma_k).
\]  

(2)

The \(p\)-value (2) is the probability that the \(r\)-th order statistic for random data under the null would be less or equal to the value of the \(r\)-th order statistic for the observed data.

We now offer some remarks regarding how our multiple testing procedure differs from those typically used when the number of hypotheses is large (say, in the thousands). To analyze gene expression data, it is often appropriate to employ a joint null distribution \(Q\) that allows for dependencies among the genes. These dependencies are unknown, so an estimate of the null distribution of the test statistics (for the cyclohedron test, the vector of counts) is made from bootstrap samples of the data; two such estimates are the null shift and scale [Birkner et al. (2005)] and null quantile [van der Laan and Hubbard (2006)] distributions. Further, there is typically one null hypothesis per gene, and \(p\)-values are assigned to control some general Type-I error rate. However, our chosen joint null distribution \(Q_0\) is simple and can be computed exactly. In addition, this test was motivated by the exploratory data analysis described in the next section. Having a powerful procedure was not critical; rather, the aim was to identify groups of top genes for further biological testing. In other settings, however, different choices of joint null distribution or of multiple testing procedure (such as those in [Birkner et al. (2005)]) can improve power.

### 4 Application to mouse microarray data

We applied the cyclohedron test to microarray data from recent work that investigated the mouse segmentation clock [Dequéant et al. (2006)]. Dequéant et al. took 17 expression measurements clock [Dequéant et al. (2006)]. Dequéant et al. took 17 expression measurements from mouse presomitic mesoderm on Affymetrix MOE430A arrays. By independently measuring the expression of the gene Lunatic Fringe (Lfgn) which is known to be periodic within the somitogenesis cycle of embryonic development, Dequéant et al. ordered the 17 experiments within the cycle. Each array consisted of over 22,000 probesets,
however we restrict the analysis to a subset of 13,873 probesets by removing genes whose expressions are deemed “absent” across the experiments by Affymetrix standards. In other words, the data consisted of 13,873 data vectors \( v \), each of which was the expression level of one gene (divided by the mean across experiments and transformed to \( \log_2 \)). We then applied the cyclohedron test to these data. We were interested in those genes whose counts \( c(v) \) were small. Accordingly, we ranked the genes by their counts; Table 1 presents the first 32 genes. The gene descriptions appear in Table 2. Table 3 lists the significance of top groups of genes. For example, the first 32 genes collectively have a p-value of 0.081, which suggests that these 32 genes are of interest.

At this point we recall the definition of a p-value which was given in equation (2). The p-value of the rank-1 gene \( \text{Obox1} \) is the probability under the null hypothesis that the top-ranked permutation count is less than or equal to 480, while the p-value of the first 16 genes (the number 0.008 in Table 3) is the probability that the gene ranked 16 has permutation count less than or equal to 4928. It is important to emphasize that the p-values do not reveal the significance of any individual gene, but rather of a collection of genes. For example, the top 19 genes having a collective p-value of 0.046 means this: the probability that the first 19 genes would collectively all have permutation count at most 6825 under the null distribution is 0.046. In other words, the group as a whole is significant. However, we determine whether any individual gene in that group is significant. For example, there is no significance to the fact that \( \text{Obox1} \) is ranked first. While it appears to be the most periodic pattern in the data by our analysis, that could have happened by chance (p-value 0.279). A natural cutoff value is to look at the first 32 genes because collectively they have a p-value of 0.081 (the next ten p-values are between 0.13 and 0.30). Note that analyses of microarray data have this property, that Type-1 errors are all but guaranteed due to the large number of genes (and thus the large number of hypotheses) that are tested. Our computations were performed with the statistical software R [R Team (2005)], using the implementation described in the Appendix.

Dequéant et al. performed significance testing according to a Lomb-Scargle analysis, and then based on gene expression profile clustering, they identified genes belonging to three pathways Notch/FGF and Wnt that are involved with somitogenesis. There are genes that are deemed interesting by both the analysis of Dequéant et al. and the cyclohedron test. For example, \( \text{Axin2} \) is ranked highly by the Lomb-Scargle (rank 6) and the cyclohedron test (rank 5). In addition, \( \text{nrarp} \) (rank 4 according to the cyclohedron test) is ranked poorly by Lomb-Scargle (rank 482), although it belongs to the Notch pathway and its gene expression clusters accordingly. Finally, there are novel genes
Table 1: The 32 genes ranked highest by the cyclohedron test. The suffix “.at” was removed from each ProbeSet ID.

<table>
<thead>
<tr>
<th>Rank</th>
<th>ProbeSet</th>
<th>Gene Name</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1456017_x</td>
<td>Obox1</td>
<td>480</td>
</tr>
<tr>
<td>2</td>
<td>1452041</td>
<td>Klhl26</td>
<td>1440</td>
</tr>
<tr>
<td>3</td>
<td>1418593</td>
<td>Taf6</td>
<td>1560</td>
</tr>
<tr>
<td>4</td>
<td>1417985</td>
<td>Nrarp</td>
<td>1950</td>
</tr>
<tr>
<td>5</td>
<td>1436845</td>
<td>Axin2</td>
<td>2240</td>
</tr>
<tr>
<td>5</td>
<td>1436343</td>
<td>Chd4</td>
<td>2240</td>
</tr>
<tr>
<td>7</td>
<td>1426267</td>
<td>Zbtb8os</td>
<td>2310</td>
</tr>
<tr>
<td>8</td>
<td>1420360</td>
<td>Dkk1</td>
<td>2520</td>
</tr>
<tr>
<td>9</td>
<td>1449643</td>
<td>Btf3</td>
<td>2772</td>
</tr>
<tr>
<td>10</td>
<td>1417399</td>
<td>Gas6</td>
<td>2800</td>
</tr>
<tr>
<td>11</td>
<td>1418102</td>
<td>Hes1</td>
<td>3120</td>
</tr>
<tr>
<td>12</td>
<td>1448799</td>
<td>Mrps12</td>
<td>3150</td>
</tr>
<tr>
<td>13</td>
<td>1418729</td>
<td>Star</td>
<td>3600</td>
</tr>
<tr>
<td>14</td>
<td>1425424</td>
<td>MGC7817</td>
<td>3850</td>
</tr>
<tr>
<td>15</td>
<td>1455740</td>
<td>Harpa1</td>
<td>4004</td>
</tr>
<tr>
<td>16</td>
<td>1450204_a</td>
<td>Mynn</td>
<td>4928</td>
</tr>
<tr>
<td>17</td>
<td>1449120_a</td>
<td>Pcm1</td>
<td>6006</td>
</tr>
<tr>
<td>18</td>
<td>1423106</td>
<td>Ube2b</td>
<td>6720</td>
</tr>
<tr>
<td>19</td>
<td>1420386</td>
<td>Seh11</td>
<td>6825</td>
</tr>
<tr>
<td>20</td>
<td>1456380_x</td>
<td>Cnn3</td>
<td>8008</td>
</tr>
<tr>
<td>21</td>
<td>1419438</td>
<td>Sim2</td>
<td>8640</td>
</tr>
<tr>
<td>22</td>
<td>1426524</td>
<td>Gnpda2</td>
<td>9009</td>
</tr>
<tr>
<td>23</td>
<td>1438557_x</td>
<td>Dnpep</td>
<td>9450</td>
</tr>
<tr>
<td>24</td>
<td>1454904</td>
<td>Tm1</td>
<td>10500</td>
</tr>
<tr>
<td>25</td>
<td>1448951</td>
<td>Tnfrsf1b</td>
<td>10530</td>
</tr>
<tr>
<td>25</td>
<td>1433952</td>
<td>Tufm</td>
<td>10530</td>
</tr>
<tr>
<td>27</td>
<td>1422327_s</td>
<td>G6pd2/</td>
<td>10725</td>
</tr>
<tr>
<td></td>
<td></td>
<td>G6pdx</td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>1416295_a</td>
<td>Il2rg</td>
<td>10920</td>
</tr>
<tr>
<td>29</td>
<td>1417316</td>
<td>Them2</td>
<td>11025</td>
</tr>
<tr>
<td>30</td>
<td>1450242</td>
<td>Tlr5</td>
<td>11232</td>
</tr>
<tr>
<td>31</td>
<td>1449164</td>
<td>Cd68</td>
<td>11340</td>
</tr>
<tr>
<td>32</td>
<td>1418337</td>
<td>RpiA</td>
<td>11760</td>
</tr>
<tr>
<td>Rank</td>
<td>Gene Name</td>
<td>Gene Description</td>
<td></td>
</tr>
<tr>
<td>------</td>
<td>-----------</td>
<td>------------------</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>Obox1</td>
<td>similar to oocyte specific gene</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Klhl26</td>
<td>kelch-like 26 (Drosophila)</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Taf6</td>
<td>TAF6 RNA polymerase II</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Nrarp</td>
<td>Notch-regulated ankyrin repeat protein</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Axin2</td>
<td>axin2</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Chd4</td>
<td>chromodomain helicase DNA binding protein</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Zbtb8os</td>
<td>zinc finger and BTB domain</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Dkk1</td>
<td>dickkopf homolog 1 (Xenopus laevis)</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>Btf3</td>
<td>basic transcription factor 3</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>Gas6</td>
<td>growth arrest specific 6</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>Hes1</td>
<td>hairy and enhancer of split 1 (Drosophila)</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>Mrps12</td>
<td>mitochondrial ribosomal protein S12</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Star</td>
<td>steroidogenic acute regulatory protein</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>MGC7817</td>
<td>hypothetical protein LOC620031</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>Hnrrpa1</td>
<td>heterogeneous nuclear ribonucleoprotein</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>Mynn</td>
<td>myoneurin</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>Pcm1</td>
<td>pericentriolar material 1</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>Ube2b</td>
<td>ubiquitin-conjugating enzyme E2B</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>Seh11</td>
<td>SEH1-like (S. cerevisiae)</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>Cnn3</td>
<td>calponin 3, acidic</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>Sim2</td>
<td>single-minded homolog 2 (Drosophila)</td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>Gnepd2</td>
<td>glucosamine-6-phosphate deaminase 2</td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>Dnpep</td>
<td>aspartyl aminopeptidase</td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>Tmnt1</td>
<td>X-linked myotubular myopathy gene 1</td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>Tnfrsf1b</td>
<td>tumor necrosis factor receptor superfamily</td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>Tufm</td>
<td>Tu translation elongation factor</td>
<td></td>
</tr>
<tr>
<td>27</td>
<td>G6pd2/</td>
<td>glucose-6-phosphate dehydrogenase 2</td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>G6dza</td>
<td></td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>Il2rg</td>
<td>interleukin 2 receptor, gamma chain</td>
<td></td>
</tr>
<tr>
<td>29</td>
<td>Them2</td>
<td>thioesterase superfamily member 2</td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>Thr5</td>
<td>toll-like receptor 5</td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>Cd68</td>
<td>CD68 antigen</td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>RpiA</td>
<td>ribose 5-phosphate isomerase A</td>
<td></td>
</tr>
</tbody>
</table>

Table 2: The gene descriptions for the 32 highest ranked genes are derived from those provided by Affymetrix.
such as Obox (rank 1 by the cyclohedron test, but not known to be related to somitogenesis) that require further investigation. This suggests that to find periodic gene expression, it is beneficial to apply many methods, including Lomb-Scargle, clustering, and the cyclohedron test. Doing so enables us to find genes overlooked by each method, as well as to confirm findings of other tests. In other words, the findings of each method complement those of others by identifying candidate genes for knockout experiments. The forthcoming paper [Dequéant et al. (2007)] will compare various methods, including Lomb-Scargle, up-down analysis, and the cyclohedron test, for identifying cyclic genes from this data set.

In conclusion, we remark that, although microarray expression analyses are often criticized due to the noise in individual measurements, the massively parallel nature of the experiments provides the possibility for finding groups of significant genes. Indeed, we confirm this in our analysis of the Dequéant et al. experiments [Dequéant et al. (2006)], in which we are unable to confirm whether any individual gene is statistically significant, yet we can identify a group of genes that collectively are significant. The biological significance of an individual gene can be determined by further targeted experimental validation.

### 5 Combinatorics of the cyclohedron test

We now describe the combinatorics and geometry behind our test. First, the set $C_n$ of cyclic signatures is in natural bijection with the vertices of a certain convex polytope. The $n$-cycle has $n(n-1)$ connected induced proper subgraphs, namely, the cyclic segments of the form $S = \{i, i + 1, \ldots, i + k\}$. Here $k < n-1$, and the indices are understood modulo $n$. The cyclohe-
**dron vertex** of a data vector \( v \in \mathbb{R}^n \) is the vector \( \tau(v) \in \mathbb{N}^n \) whose \( i \)-th coordinate \( \tau(v)_i \) is the number of cyclic segments \( S \) containing \( i \) such that \( v_i = \min \{ v_s : s \in S \} \). The **cyclohedron** \( C_n \) is the convex hull in \( \mathbb{R}^n \) of all the cyclohedron vertices \( \tau(v) \) where \( v \) ranges over \( \mathbb{R}^n \). For \( n = 4 \) and the data vector \( v = (0.49, 5.73, 4.01, 2.67) \), we have \( \tau(v) = (6, 1, 2, 3) \), while for \( v' = (0.49, 5.73, 2.67, 4.01) \) we have \( \tau(v') = (6, 1, 4, 1) \). For example, \( \tau(v)_3 = 2 \) because \( v_3 = 4.01 \) is minimal in \( S^1 = \{ 3 \} \) and \( S^2 = \{ 2, 3 \} \).

Two vectors in \( \mathbb{R}^4 \) share the same signature \( \sigma = \{ \sigma_1, \sigma_2, \sigma_3 \} \) if and only if they are mapped to the same cyclohedron vertex \( \tau \). The convex hull of all cyclohedron vertices \( \tau(v) \) is the 3-dimensional cyclohedron \( C_4 \). This is a simple polytope with 20 vertices, 30 edges and 12 facets (for the 12 cyclic segments). It is depicted in Figure 3. Vertices in the figure, incident to a \('dual'\) edge indicate signatures \( \sigma \) with \( c(\sigma) = 2 \). Thus the set \( C_4 \) of all signatures has 20 elements, one for each vertex of \( C_4 \).

![Diagram](image.png)

Figure 4: The cyclohedron \( C_4 \); its vertices correspond to the distinct signatures for \( n = 4 \). Following the notation of [Hemmecke et al. (2006)], the string \( \frac{1}{3}24 \) labels the cyclohedron vertex of data vectors whose descent permutation is 1324 or 3124.

The following theorem summarizes what is known about the cyclohedron. It is extracted from [Fomin and Reading (2004), Hohlweg and Lange (2005),...
Markl (1999)].

**Theorem 5.1.** The cyclohedron $C_n$ is an $(n-1)$-dimensional polytope. It is the solution set in $\mathbb{R}^n$ of the following system of one linear equation and $n(n-1)$ linear inequalities:

$$x_1 + x_2 + \cdots + x_n = n(n-1), \quad (3)$$

$$\sum_{s \in S} x_s \geq \left( \frac{|S|+1}{2} \right) \text{ for each cyclic segment } S. \quad (4)$$

The cyclohedron $C_n$ is simple, i.e. each vertex lies on precisely $n-1$ facets. Each inequality (4) defines a facet. The total number of vertices equals $\binom{2n-2}{n-1}$. More generally, the number $f_i$ of $i$-dimensional faces of the cyclohedron is given by the generating function

$$\sum_{i=0}^{n-1} f_i \cdot z^i = \sum_{k=0}^{n-1} \binom{n-1}{k}^2 \cdot (z+1)^k. \quad (5)$$

Algorithm 2.1 is a greedy method for linear programming on the cyclohedron $C_n$. Indeed, computing the cyclohedron vertex $\tau(v)$ of a data vector $v = (v_1, \ldots, v_n)$ is equivalent to the linear program of minimizing $\sum_{i=1}^{n} v_i x_i$ subject to the constraints (3) and (4). The optimal vertex of that linear program on $C_n$ is precisely the vector $x = \tau(v)$.

Given the linear functional $\sum_{i=1}^{n} v_i x_i$ to be minimized, Algorithm 2.1 generates a collection $\sigma = \{\sigma_1, \sigma_2, \ldots, \sigma_{n-1}\}$ of subsets of $\{1, 2, \ldots, n\}$. These sets $S = \sigma_i$ are cyclic segments, and they indicate which $n-1$ inequalities (4) are tight at the optimal vertex $x = \tau(v)$ of $C_n$. This implies that $\tau(v)$ can be recovered from $\sigma(v)$ and vice versa:

**Corollary 5.2.** The cyclohedron vertex $\tau(v)$ of any data vector $v \in \mathbb{R}^n$ can be obtained from the signature $\sigma(v) = \{\sigma_1, \sigma_2, \ldots, \sigma_{n-1}\}$ by solving the linear system of equations

$$(3) \quad \text{and} \quad \sum_{s \in \sigma_i} x_s = \left( \frac{|\sigma_i|+1}{2} \right) \text{ for } i = 1, 2, \ldots, n-1. \quad (6)$$

Conversely, the signature $\sigma(v)$ is recovered from the vertex $\tau(v)$ by substituting $x = \tau(v)$ into the inequalities (4) and collecting all index sets $S$ for which equality holds.
In light of Corollary 5.2, we henceforth shall identify signatures $\sigma \in C_n$ with their corresponding vertices $\tau$ of the cyclohedron $C_n$. We note that the solution $\tau$ to (6) can be read off easily within Algorithm 2.1. It always holds that $\tau_{\delta_n} := \binom{n}{2}$, and the other $n-1$ coordinates are obtained by adding one line at the end of the main $i$ loop:

**Output** $\tau_{\delta_i} = (|\text{Left}| + 1) \cdot (|\text{Right}| + 1)$.

Two data vectors $v$ and $v'$ are *cyclically equivalent* if and only if $\sigma(v) = \sigma(v')$, i.e., if and only if the linear functionals corresponding to $v$ and $v'$ are minimized at the same vertex $\tau(v) = \tau(v')$ of the cyclohedron $C_n$. The cyclic equivalence classes are the normal cones at the vertices of $C_n$. They are specified by the inequalities $v_{\delta_i} < v_{\delta_h}$ for all inclusions $\sigma_k \subset \sigma_j$ in $\sigma(v)$. Since $C_n$ is simple, $n-1$ inclusions suffice, and these can be generated by augmenting Algorithm 2.1, again at the end of the main $i$ loop, as follows:

```
if Right $\neq \emptyset$ or Left $\neq \emptyset$ then output $v_{\delta_i} < v_{\delta_h}$.
```

The generated inequalities permit the study of confidence regions for the cyclohedron test.

**Example 5.3.** Fix $n = 17$ and let $v$ be the data vector for the 0box gene in Example 2.4. The augmented Algorithm 2.1 reveals that the cyclic equivalence class of $v$ is given by

$$v_{11} < v_{10} < v_9 < v_8 < v_6 < v_5 < v_{12} < v_{14} < v_{15} < v_{17} < v_4 < v_3 < v_1 < v_2$$
and
$$v_6 < v_7$$
and
$$v_{17} < v_{16}$$
and
$$v_{14} < v_{13}.$$

These inequalities specify the normal cone at the vertex

$$\tau(v) = (2, 1, 3, 4, 11, 24, 1, 14, 15, 16, 136, 10, 1, 16, 7, 1, 10)$$

of the 16-dimensional cyclohedron $C_{17}$. Recall that the possible signatures for data with $n = 17$ are (in bijection with) the vertices of $C_{17}$, and their total number equals

$$|C_{17}| = \binom{2 \cdot 17 - 2}{17 - 1} = \binom{32}{16} = 601,080,390.$$

Among all these signatures, the vertex $\tau(v)$ is of interest because the probability that a random linear functional attains its minimum over $C_{17}$ at that vertex is rather small:

$$p(v) = \frac{c(v)}{n!} = \frac{480}{17!} = 1.35 \cdot 10^{-12}.$$ 

The results of our analysis for the full data set were presented in Section 4. □
The theory of graph associahedra also offers the following combinatorial characterization of the possible outputs of Algorithm 2.1. A collection \( \{\sigma_1, \sigma_2, \ldots\} \) of cyclic segments is called a tubing of the \( n \)-cycle if any two elements satisfy the following property: either \( \sigma_i \subset \sigma_j \), or \( \sigma_j \subset \sigma_i \), or \( \sigma_i \) and \( \sigma_j \) are disjoint and no node in \( \sigma_i \) is adjacent to a node in \( \sigma_j \). Each maximal tubing has the same number of elements, namely \( n-1 \), and the maximal tubings are precisely the signatures generated by Algorithm 2.1. The simplicial complex of all tubings is dual to the face poset of the simple polytope \( C_n \). Analogous statements hold for the face poset of the graph associahedron of any graph \( G \) with vertex set \( \{1, 2, \ldots, n\} \). The cyclohedron \( C_n \) is the special case when \( G \) is the \( n \)-cycle.

We propose that rank tests which are associated with graphs \( G \) in this manner be called topographical models. This is motivated by their relationship with graphical models (Markov random fields) which was developed in [Morton et al. (2007)]. Our cyclohedron vertex map \( \tau \), for \( G \) the \( n \)-cycle, was denoted \( \tau_{K_G}^* \) in [Morton et al. (2007), Section 5]. We believe that topographical models for graphs \( G \) other than the \( n \)-cycle will be useful for wide range of statistical problems concerning data with an underlying graphical structure.

6 Null distribution of the counts and computing order statistics

We next compute two probability distribution functions, that of the random variable \( c \) and of its order statistics. In the first part of this section we introduce a generating function that represents the distribution \( P_c \) of \( c \) under the null distribution. This is applied in the second part to derive the order statistics of \( P_c \) and a formula for computing the collective p-values (2) exactly. Recall that the set \( C_n \) of signatures equals the set of maximal tubings or vertices of the cyclohedron \( C_n \). For each \( \sigma \in C_n \), the quantity \( c(\sigma) = p(\sigma) \cdot n! \) is the number of permutations \( \delta \) which map to \( \tau \). See Algorithm 2.1 and Proposition 2.2.

We define the count generating function for the cyclohedron test to be the following polynomial:

\[
\Gamma_n(t) := \sum_{\sigma \in C_n} t^{c(\sigma)}.
\]

By Theorem 5.1, this polynomial gives a refinement of the central binomial coefficient:

\[
\Gamma_n(1) = |\text{Vert}(C_n)| = \binom{2n-2}{n-1}.
\]
Similarly, the first derivative $\Gamma'_n(t) = \frac{d}{dt} \Gamma_n(t)$ gives a refined count of the permutations:

$$\Gamma'_n(1) = |\Sigma_n| = n!.$$ 

We list the first few non-trivial instances of the count generating function:

\[
\begin{align*}
\Gamma_4(t) &= 4t^2 + 16t, \\
\Gamma_5(t) &= 20t^3 + 10t^2 + 40t, \\
\Gamma_6(t) &= 12t^8 + 24t^6 + 48t^4 + 48t^3 + 24t^2 + 96t, \\
\Gamma_7(t) &= 28t^{20} + 56t^{15} + 140t^{10} + 28t^8 + 56t^6 + \cdots + 112t^3 + 56t^2 + 224t, \\
\Gamma_8(t) &= 8t^{80} + 32t^{48} + 128t^{45} + 64t^{40} + 64t^{36} + \cdots + 256t^3 + 128t^2 + 512t, \\
\Gamma_9(t) &= 72t^{210} + 72t^{168} + 108t^{140} + 144t^{126} + \cdots + 576t^3 + 288t^2 + 1152t.
\end{align*}
\]

The count generation function encodes the probability distribution function of $c$ in the following way:

**Remark 6.1.** The probability $P_c(\gamma)$ is the coefficient of $t^\gamma$ in the polynomial $(t/n!) \cdot \Gamma'_n(t)$.

**Example 6.2.** Consider the case $n = 7$. The $s_7 = 10$ possible permutation counts are

$$\text{im}(c) = \{1, 2, 3, 4, 5, 6, 8, 10, 15, 20\}.$$ 

The probability for each of these counts to be observed is the corresponding coefficient in

$$\sum_{\gamma \in \text{im}(c)} P_c(\gamma) \cdot t^\gamma = \frac{t}{5040} \cdot \Gamma'_7(t) = \frac{1}{9} t^{20} + \frac{1}{6} t^{15} + \frac{5}{18} t^{10} + \cdots + \frac{1}{45} t^2 + \frac{2}{45} t.$$ 

For instance, the cyclohedron $C_6$ has 56 vertices $\sigma$ with $c(\sigma) = 15$, and this accounts for $56 \cdot 15 = 840$ of the 5040 permutations $\delta$ in $\Sigma_7$. Thus the probability that a random data vector $\nu \in \mathbb{R}^7$ has permutation count $c(\nu) = 15$ is equal to $P_c(15) = 840/5040 = 1/6$.

We now describe a formula for computing the count generating function. Let $\mathcal{T}_m$ denote the set of unlabeled rooted trees with $m$ nodes, where each node has at most two children. The number of these trees is the *Wedderburn-Etherington number*, denoted by $t_m := |\mathcal{T}_m|$. Starting with $t_0 = 1$, the Wedderburn-Etherington numbers are

$$1, 1, 2, 3, 6, 11, 23, 46, 98, 207, 451, 983, 2179, 4850, 10905, 24631, 56011, \ldots$$
and they can be computed by the following recursion:

\[
    t_m = \sum_{i=0}^{[m/2]-1} t_i \cdot t_{m-i-1} \quad \text{if } m \text{ is even,} \\
    t_m = \left(t^{(m-1)/2}_m\right) + \sum_{i=0}^{[m/2]-1} t_i \cdot t_{m-i-1} \quad \text{if } m \text{ is odd.}
\]

This holds because each tree \( T \) in \( T_m \) is constructed uniquely by taking an unordered pair consisting of a tree \( T_1 \) in \( T_i \) and a tree \( T_2 \) in \( T_{m-i-1} \) and attaching them to a new root. Note that \( t_0 = 1 \) corresponds to the case when the new root has outdegree one. We call \( \binom{m-1}{i} \) the order of the root. The node is called balanced if \( i = (m-1)/2 \) and the two subtrees \( T_1 \) and \( T_2 \) are isomorphic. In this manner, each node of a tree \( T \in T_m \) has an order, and it is either balanced or unbalanced. For instance, all leaves are balanced of order 1, all nodes with one child are unbalanced of order 1, and nodes with two children have order \( \geq 2 \). For a tree \( T \in T_m \) let \( \text{unbal}(T) \) denote the number of unbalanced nodes in the tree, and let \( \text{order}(T) \) denote the product of the orders of all nodes in \( T \).

**Theorem 6.3.** The count generating function for the cyclohedron test equals

\[
    \Gamma_n(t) = n \cdot \sum_{T \in T_{n-1}} 2^{\text{unbal}(T)} \cdot t^{\text{order}(T)}.
\]

**Proof.** Every signature \( \sigma = \{\sigma_1, \ldots, \sigma_{n-1}\} \) in \( \mathcal{C}_n \) maps to an unordered tree \( T = T(\sigma) \) in \( T_{n-1} \). If \( n = 2 \) then \( T \) is the tree with one node. For \( n \geq 3 \) we construct \( T \) iteratively as in Algorithm 2.1: by induction, the sets Left and Right correspond to two subtrees \( T_1 \) and \( T_2 \), and a new root is attached to form the tree corresponding to \( \{\delta_i\} \cup \text{Right} \cup \text{Left} \). The order of the resulting tree \( T(\sigma) \) equals the permutation count \( c(\sigma) \) computed. It remains to be shown that the set of all signatures \( \sigma \) which are mapped to the same tree \( T \in T_{n-1} \) has precisely \( n \cdot 2^{\text{unbal}(T)} \) elements. The factor \( n \) comes from the fact that the last element \( \delta_n \) can be chosen arbitrarily. So, let us suppose \( \delta_n = n \). Then the indices appearing in \( \sigma \) are precisely \( 1, 2, \ldots, n-1 \). Let \( T_1 \) and \( T_2 \) be the two subtrees of the root of \( T \), and suppose they have \( i \) and \( n-2-i \) nodes respectively. If \( i \neq n/2 \) then either \( \delta_{n-1} = i+1 \) and both \( \{1, 2, \ldots, i\} \) and \( \{n-2-i, \ldots, n-2, n-1\} \) are in \( \sigma \), or \( \delta_{n-1} = n-1-i \) and both \( \{1, 2, \ldots, n-2-i\} \) and \( \{n-i, \ldots, n-2, n-1\} \) are in \( \sigma \). If \( i = n/2 \) then \( \delta_{n-1} = n/2 \) and both \( \{1, 2, \ldots, n/2-1\} \) and \( \{n/2+1, \ldots, n-1\} \) are in \( \sigma \). The choices for the remaining elements of \( \sigma \) are constructed inductively by identifying the nodes of the two subtrees with these two sets. If the two subtrees are identical (i.e. the root is balanced) then there is only one identification to be
considered, otherwise we must consider two cases. Proceeding in this manner along the tree, we see that there are $2^{\text{order}(T)}$ many choices of signatures $\sigma$ on \{1, 2, \ldots, n - 1\} which map to $T$.

We next present a recursive method for computing the count generating function $\Gamma_n(t)$. Let $f = \sum_i a_i t^i$ and $g = \sum_j b_j t^j$ be any two generating functions and $M$ any positive integer. Then we define the *-product of $f$ and $g$ with respect to $M$ as follows:

$$f*_{M}g := \sum_{i,j} a_i \cdot b_j \cdot t^{i+jM}. \quad (7)$$

**Corollary 6.4.** Let $\Omega_n(t)$ be the polynomial defined recursively by

$$\Omega_0(t) = \Omega_1(t) = t \quad \text{and} \quad \Omega_m(t) = \sum_{i=0}^{m-1} \Omega_i(t) *_{(m-1)} \Omega_{m-1-i}(t).$$

Then the polynomial $\Gamma_n(t) = n \cdot \Omega_{n-1}(t)$ is the count generating function for the cyclohedron test.

**Proof.** This follows from the recursive tree construction in the proof of Theorem 6.3. \hfill \Box

**Example 6.5.** Corollary 6.4 easily yields the full expansion of $\Omega_n(t)$ for small values of $n$. For $n = 17$, the case of interest in Section 4 (see also Examples 2.4 and 5.3), we find

$$\Gamma_{17}(t) = 272t^{108108000} + 544t^{80689600} + 272t^{86486400} + 544t^{80720640} + \cdots + 417792t^6 + 278528t^5 + 278528t^4 + 278528t^3 + 139264t^2 + 557056t.$$

The number of terms in this polynomial equals $|\text{im}(c)| = 2438$. The 2438 values of the probability distribution function $P_c$ are plotted on a logarithmic scale in Figure 4. For larger values of $n$, say $n \geq 30$, it becomes infeasible to compute the expansion of $\Gamma_n(t)$, but Corollary 6.4 can still be used to design efficient methods for sampling from $P_c$. \hfill \Box

The distribution function $F_{(r)}(\gamma)$ of the order statistic $C_{(r)}$ is now computed. Defining $p_i = P_c(\gamma_i)$ to be the probability under the null hypothesis that the count is equal to $\gamma_i$, Remark 6.1 tells us that

$$(t/n!) \cdot \Gamma_n'(t) = p_1 t^{\gamma_1} + p_2 t^{\gamma_2} + \cdots + p_n t^{\gamma_n}, \quad \text{where} \quad \gamma_1 < \gamma_2 < \cdots < \gamma_n.$$
Consider the identity
\[(p_1 + p_2 + \cdots + p_n)^N = \sum_{i_1+i_2+\cdots+i_n=N} \binom{N}{i_1 i_2 \cdots i_n} p_1^{i_1} p_2^{i_2} \cdots p_n^{i_n} = 1.\]

By definition, the distribution function \(F_r(\gamma_k)\) is the sub-sum of all the terms in this sum whose indices satisfy
\[i_1 + \cdots + i_{k-1} < r \leq N - i_{k+1} - \cdots - i_n.\]

For the purpose of computational efficiency we rewrite this sub-sum as follows. The formula below furnishes us with an efficient method for computing the collective p-values.

**Lemma 6.6.** The probability distribution function \(F_r(\gamma_k)\) of the order statistic \(C_r\) under the null distribution \(Q_0\) is given by the formula
\[
\sum_{i=1}^{N} \sum_{j=\max(0, r-i)}^{\min(r-1, N-i)} \binom{N}{i, j, N-i-j} (p_1 + \cdots + p_{k-1})^j \cdot p_k^i \cdot (p_{k+1} + \cdots + p_n)^{N-i-j}.
\]

**Proof.** The first sum is over the number of data points that have permutation count \(\gamma_k\). The second sum is over \(j\), the number of data points whose permutation count is less than \(\gamma_k\). Then, the multinomial coefficient gives the possible ways to partition \(\{1, 2, \ldots, N\}\) into sets of size \(i, j, N-i-j\); that is, it accounts for possible rearrangements among the permutation counts equal to, less than, and greater than \(\gamma_k\). The probability that such rearrangement occurs is the product \((p_1 + \cdots + p_{k-1})^j \cdot p_k^i \cdot (p_{k+1} + \cdots + p_n)^{N-i-j}\). \(\square\)

**Appendix: R code for the cyclohedron test**

The R source code `topoGraph.R` is available for the cyclohedron test. The software can be downloaded from

http://bio.math.berkeley.edu/ranktests/index.html

Our code requires the free statistical software package R [R Team (2005)]. Here we describe how to perform basic tasks related to the cyclohedron test. The data file must be a CSV (comma-separated values) file, where the first column consists of identifying labels (such as gene names), and the first row labels the time points (all other rows are the corresponding data vectors). We illustrate the use of the basic functions with the data file (named ‘13873.csv’).
that we described in Section 4. The first column consists of the ProbeSet IDs. The source code containing the R functions is `topoGraph.R`. First, we call the source code and load the data file from an R command line (here, we assume that both files are in the current working directory):

```r
source("topoGraph.R")
dataset<-loaddata("13873.csv")
```

Next, we calculate the count of each data vector, using the following command:

```r
counts<-cycleCounts(dataset)
```

This defines “counts,” a vector which lists the counts of the data vectors in the order given by the data file. To list the genes according to their count ranking, as shown in Table 1, we call the function `rankby` which outputs the labels (here, the ProbeSet IDs) of the genes. The following command outputs the ten highest ranked ProbeSets.

```r
rankby(row.names(dataset),counts)[1:10]
```

```r
[1] "1456017_x_at"  "1452041_at"  "1418593_at"  "1417985_at"
[5] "1436845_at"  "1436343_at"  "1426267_at"  "1420360_at"
[9] "1449643_s_at"  "1417399_at"
```

More extensive documentation is available online.

References


