Chapter 1

Statistical Design of an Experimental Problem in Harmonics

Rita Aggarwala[1], Jahrul M. Alam[2], Md. Shafiqul Islam[3], Michael Lamoureux[1], Marc Paulhus[1], Miro Powojowski[1], Leila Rasekh[4]
Report prepared by M. Lamoureux

1.1 Abstract

The Michelin Tire Company requires its tires to be very uniform in order to provide a smooth, quiet ride. The design of the appropriate manufacturing technique leads to a problem in harmonic analysis, and the design of a statistical experiment to accurately measure the harmonic components contributing to a measured force on the tire. Optimal designs were developed, as well as a number of useful Monte Carlo methods. MATLAB code of the tests are provided.

1.2 Introduction

The Michelin Tire Company is interested in manufacturing tires that meet a certain level of uniformity. Generally speaking, the more uniform the tire, the smoother the ride and the quieter is the tire, in terms of rolling noise perceived by the passengers. Indeed, many of the contracts Michelin holds for tire production require that they manufacture tires that meet certain specified standards of uniformity.

Uniformity of a tire is evalued by measuring the force exerted by the tire on a measuring device, as the tire is rotated 360 degrees about an axle. The force may be measured in the radial direction, in the axial direction, or both. This produces one or two force curves which describe the uniformity of the tire: a flat curve indicates a perfectly uniform tire. A non-flat curve can be expressed as a sum of sines and cosines, and the manufacturer is interested only

[1]University of Calgary
[2]University of Alberta
[3]Concordia University
[4]University of Guelph

in the magnitude of specific harmonics of this force curve. For instance, a contract may require that the manufactured tires have harmonics one through five with magnitudes below a certain threshold. Thus the problem of ensuring uniformity of the tires is more precisely a problem of minimizing, or reducing below a certain threshold, certain harmonic components of one or two force curves.

To understand how the non-uniformities in a tire appear, and how one may control them, it is necessary to understand the basic construction of a tire. Each tire is made up of a series of components, or sheets of material, which are wrapped one layer on top of another, while stretched out over a tire mold. For instance, the first layer would be an inner, airtight sheet of reinforced rubber which will form the inner air compartment for the tire. Next comes several layers of different types of rubber, then layers of cords and/or steel belting for reinforcement, more rubber, and eventually the treads are laid on top to finish off the tire.

There may be as many as twenty layers built up into the tire. As each sheet of material is wrapped around the mold, the ends are joined by glue or by melting, which results in a small bead of material at the join. These beads will lead to bumps, or non-uniformities in the final tire assembly. There are other factors as well that cause each layer to contribute some non-uniformities to the final product; the causes are not in question here, but they are indicated so it is understood that each layer somehow contributes to the uniformity or non-uniformity of the tire.

The technician has little control in what causes the particular bumps or non-uniformity in the layers that go into building the tire. One thing that the technician can control is the position (starting angle of assembly) for each layer going into the tire. For instance, the first layer may be aligned at 0 degrees relative to the mold, second layer rotated 15 degrees from the first layer, third layer rotated at 30 degrees, and so on.

The technician would like to choose this series of rotations in order to produce a tire that meets the uniformity requirements. This leads to a canonical problem of analysis and synthesis. One must first analyse the layers to see how each one contributes to the final uniformity, or non-uniformity, of the finished tire; then one must use this knowledge to design the finished tire that meets the specified uniformity requirements. The only freedom one has in both the analysis and synthesis is choosing the position of each layer.

Our goal in this problem is to solve the analysis problem. The task is to design an experiment where a technician can construct a series of test tires that will be used to determine, with statistical confidence, the contribution of each layer to the final profile, or force curve, of the generic tire. Once the contributions are known, the synthesis problem of finding the optimally smooth tire, or one that meets any specific uniformity conditions on the harmonic components, is relatively straightforward, and need not concern us here.

1.3 Methodology

It was clear early that this problem would require techniques of both Fourier analysis and statistics: Fourier techniques to describe the harmonic problem, and statistical techniques to deal with the problem of accurately measuring quantities that come from real experimental measurements. A team of researchers was assembled that brought in expertise from both areas.

It turned out that Monte Carlo methods would be useful as well, and appropriate simulations on the computer would be needed. MATLAB was chosen as the computational platform, as it combined the power of complex, matrix linear algebra required by the harmonic analysts with the statistical tools needed by the statisticians.

As the research progressed, our industrial collaborator from Michelin provided some very useful information based on their experience with manufacturing and analysing their tires. In particular, he indicated what types of statistical experiments had been performed, what tests looked promising, what constraints and costs could be expected in running the test. This information directed much of the research below, and details are indicated where appropriate.

A number of very promising, and even optimal, designs were obtained over the course of the workshop. Given the limited time available in a one week period, it was impossible to obtain complete characterizations for these designs, their robustness, and other features. With MATLAB, some good verifications were produced that are quite convincing. Thus, the results brought the analysis problem all the way to a solution, and our industrial collaborator has indicated it is a significant, valuable answer to their problem.

1.4 Simplifications

While the original problem involved a choice of one or two force curves to measure, it was quickly decided to work with just one force curve. This is only a minor simplification. The Monte Carlo methods developed for one curve can easily be extended to two curves. Also, in the optimal designs developed, the designs depends only on the harmonics studied under the test; thus, two curves can be included in the design by simply including the harmonics for both. Since the physical construction of test tires is expensive, it turns out this is a rather efficient way of minimizing the number of tires needed to test any number of force curves.

It was also noted that the force curves are in fact only sampled at 256 points, thus the problem of infinite dimensional spaces of curves is avoided. Indeed, a large simplification is obtained by noting a finite, discrete Fourier transform (which is easily done on the computer) reduces the harmonic problem to a natural setting.

In practice, the technician is not completely free in his choice of angles used to lay out components of the tire. In the development below, we assumed any angle could be chosen (though perhaps not all layers would be adjusted at the same time), and then one would use robustness results to see what happened when the choice of angle was restricted.

A statistical simplification is to assume the layers of the tire act independently, and additively, in producing the final force curve of the tire. This assumption is supported by the experience of the Michelin group of researchers.

Another major simplification was to get the whole team speaking a common language of Fourier series, least square approximation, and using MATLAB tools.

1.5 Fourier Series Formulation

The tire's characteristic force curve can be described as a continuous function on the interval $[0, 1]$ with periodic boundary conditions. As the problem was originally posed, it was suggested

that these periodic functions $f \in C[0,1]$ be expanded in terms of sines and cosine functions: that is, one writes

$$f(t) = \sum_{n=0}^{\infty} a_n \cos(2\pi nt) + b_n \sin(2\pi nt),$$

where a_n, b_n are real coefficients encoding the magnitude and phase of the corresponding harmonic. Algebraically, it is more convenient to use complex exponentials to expand the periodic function in a Fourier series, as

$$f(t) = \sum_{n=-\infty}^{\infty} c_n e^{2\pi int},$$

for complex coefficients c_n. Since f is real, there is some redundancy in this expansion requiring $c_{-n} = \overline{c_n}$, the complex conjugate of c_n, for each n.

More important is the redundancy due to the fact that the continuous function $f \in C[0,1]$ is observed only at finitely many points $t_k = k/256$, for $k = 0, 1, ..., 255$. The Fourier series expansion at these discrete points

$$f(t_k) = \sum_{n=-\infty}^{\infty} c_n e^{2\pi i \frac{nk}{256}}$$

collapses to a finite sum, because of the periodicity of the exponentials, so one may write a discrete Fourier expansion, with

$$f(t_k) = \sum_{n=-127}^{128} c_n e^{2\pi int_k}, \qquad t_k = 0, 1/256, 2/256, \ldots 255/256.$$

It is worth noting in passing that the Fourier coefficients c_n are quickly calculated using an FFT software routine, and so this formulation of the problem does not introduce any additional complexity in the problem.

In the problem at hand, a function f is considered the signature of a given tire component or layer that will affect the final force profile. If this component is rotated by an angle θ, the signature function f is shifted and the corresponding Fourier coefficients change. Introducting the notation S_θ for the shift operator, one obtains

$$\begin{aligned}
(S_\theta f)(t) &= f(t-\theta)) \\
&= \sum_n c_n e^{2\pi in(t-\theta)} \\
&= \sum_n (e^{-2\pi in\theta} c_n) e^{2\pi int} \\
&= \sum_n c_n^\theta e^{2\pi int}.
\end{aligned}$$

That is, the Fourier coefficients transform under the shift by θ as a linear transform $c_n \mapsto e^{-2\pi in\theta} c_n$. Equivalently, the vector of coefficients c_n transforms as

$$\begin{pmatrix} \vdots \\ c_n \\ \vdots \end{pmatrix} \mapsto \begin{pmatrix} \vdots \\ c_n^\theta \\ \vdots \end{pmatrix} = \begin{pmatrix} \ddots & & \\ & e^{-2\pi n\theta} & \\ & & \ddots \end{pmatrix} \begin{pmatrix} \vdots \\ c_n \\ \vdots \end{pmatrix}$$

which is just multiplication by a diagonal matrix D_θ whose entries are complex exponentials.

1.6 The Component Problem

A tire is built up from a number of layers (tread, cords, airtight inner rubber, etc), usually on the order of $m = 20$ components. Each layer contributes a signature $f^k(t) \in C[0,1]$ to the observed force profile

$$F(t) = \sum_{k=1}^{m} f^k(t),$$

where the assumption (based on Michelin's experience) is that the contribution is additive, and thus each layer's contribution is independent of the others. The component functions f^k cannot be measured directly; however, a factory worker may modify the construction of the tire by changing the positioning of individual layers within the test tire. Each layer may be shifted independently by some angle θ. Applying a vector of shifts $\Theta = (\theta_1, \theta_2, \ldots, \theta_m)$, where θ_k is the rotation angle for k-th layer, gives an operation on the observed force profile as

$$(S_\Theta F)(t) = \sum_{k=1}^{m} (S_{\theta_k} f^k)(t).$$

In Fourier components, this becomes

$$\begin{aligned} \sum_n C_n^\Theta e^{2\pi i n t} &= \sum_k \sum_n (e^{-2\pi i n \theta_k} c_n^k) e^{2\pi i n t} \\ &= \sum_n (\sum_k e^{-2\pi i n \theta_k} c_n^k) e^{2\pi i n t}, \end{aligned}$$

and by equating terms in the Fourier expansion one obtains the transform directly on the coefficients as

$$C_n^\Theta = \sum_{k=1}^{m} e^{-2\pi i n \theta_k} c_n^k.$$

In particular, one observes there is no mixing of harmonics: that is, the n-th harmonic of the observed (transformed) force curve is a weighted sum of the n-th harmonics of the contributing layers.

The analysis problem is to determine the coefficients c_n^k from the observed C_n^Θ, using some choice of the vector of angles Θ. Since the observed spectra are real, it is enough to consider only non-negative n in determining the harmonics, and the constant term ($n = 0$) is irrelevant. In practice, only a small values of n are of interest (eg. $n = 1, 2, \ldots, 5$), as these correspond to certain low frequency vibrations, but the coefficients must be determined for all layers (eg. $k = 1, 2, \ldots, 20$). Also note this is a statistical data problem, as the measured coefficients include measurement error and statistical deviations due to variations in the construction of these real tires.

1.7 The linear model

The problem is to determine individual coefficients c_n^k, for all layers $k = 1, \ldots, m$, from observations of the lumped coefficients C_n^Θ, where the experimental design involves choosing some appropriate vectors of angles $\Theta = (\theta_1, \theta_2, \ldots, \theta_m)$. The design also should find the number of vectors $\Theta^1, \Theta^2, \ldots, \Theta^R$ required to accurately determine the coefficients c_n^k. These coefficients must be determined for a range of harmonics, say $n = 1, \ldots, q$, and it will be convenient to design the experiments to work for all these harmonics simultaneously.

It is natural to group the coefficients into column vectors, as

$$\begin{pmatrix} c_1^k \\ \vdots \\ c_q^k \end{pmatrix} = \vec{c}^{\,k}$$

and

$$\begin{pmatrix} C_1^\Theta \\ \vdots \\ C_q^\Theta \end{pmatrix} = \vec{C}^\Theta.$$

The linear model encompassing all layers, and the range of harmonics, can thus be written in block form as

$$\begin{pmatrix} \vec{C}^{\Theta^1} \\ \vec{C}^{\Theta^2} \\ \vdots \\ \vec{C}^{\Theta^R} \end{pmatrix} = \begin{pmatrix} D_{\theta_1^1} & D_{\theta_2^1} & \ldots & D_{\theta_m^1} \\ D_{\theta_1^2} & D_{\theta_2^2} & \ldots & D_{\theta_m^2} \\ \vdots & & & \vdots \\ D_{\theta_1^R} & D_{\theta_2^R} & \ldots & D_{\theta_m^R} \end{pmatrix} \begin{pmatrix} \vec{c}^{\,1} \\ \vec{c}^{\,2} \\ \vdots \\ \vec{c}^{\,m} \end{pmatrix} + \vec{\epsilon}$$

where $\Theta^1, \ldots, \Theta^R$ is the choice of vectors of angles set in the experiment, the $D_{\theta_k^r}$ are $q \times q$ diagonal matrices with entries $e^{-2\pi i n \theta_k^r}$ on the diagonal, and $\vec{\epsilon}$ is the statistical measurement error.

More succinctly, the linear model is represented by $\vec{C}^{\vec{\Theta}} = D\vec{c} + \vec{\epsilon}$, with D a matrix in block form, each block a diagonal matrix as above. If these were real matrices, the solution via least squares is clear. It was a simple exercise, undertaken in the course of this workshop, to verify that even for complex matrices, the least square solution is obtained in a straightforward manner via computations with the usual complex inner product. Namely, one solves for $\vec{c}$ as

$$\vec{c} = (D^* D)^{-1} D^* \vec{C}^{\vec{\theta}},$$

where D^* indicates the complex conjugate transpose of the matrix D. Similarly, the variance estimates for the inversion will depend on the properties of matrix $(D^*D)^{-1}$.

Noting that D^*D is also in block form, it is convenient to permute rows and columns (essentially grouping terms by layers, rather than harmonics) to obtain matrix $X = Perm(D)$ so that X^*X is in block diagonal form, with

$$X^*X = \begin{pmatrix} (Z_1) & & & 0 \\ & (Z_2) & & \\ & & \ddots & \\ 0 & & & (Z_q) \end{pmatrix},$$

where each $m \times m$ block (Z_n) has entries

$$(Z_n)_{jk} = \sum_{r=1}^{R} e^{2\pi i n(\theta_k^r - \theta_j^r)}.$$

This greatly simplifies the analysis, since each block (Z_n) may be examined separately. Notice each such block corresponds to a separate harmonic.

The problem becomes that of estimating the regression coefficients in the multiple regression model

$$\vec{C}^{\vec{\Theta}} = X\vec{c} + \vec{\epsilon}.$$

A standard assumption is that

$$Var(\vec{\epsilon}) = \sigma^2 I_{qm}$$

for some (unknown) value σ. It was pointed out that this assumption essentially says the R tires and q harmonics act independently, and different measurements have equal error; this may be a gross oversimplification worth further investigation. For instance, there may be some bias in the way the tires are constructed for the test, or trends reflected in the sequence that the tires are built. On the other hand, the harmonics are orthogonal measures in a large dimensional space, and at least some of us were convinced that the first few harmonics would act independently, with similar measurement error. In any case, we proceed with this assumption.

The least-square estimator of $\vec{c}$ is

$$\hat{c} = (X^*X)^{-1}X^*\vec{C}^{\vec{\Theta}}$$

with variance

$$Var(\hat{c}) = (X^*X)^{-1}X^*Var(\vec{\epsilon})X(X^*X)^{-1} = \sigma^2(X^*X)^{-1}.$$

Thus the equation of finding an optimal design boils down to finding a matrix X such that X^*X is "good." For more general forms of $Var(\vec{\epsilon})$, the optimal condition is more complicated.

Some possible optimality conditions ("goodness" of X) include minimizing the determinate of matrix $(X^*X)^{-1}$ (D-optimality), minimizing the spectral norm of $(X^*X)^{-1}$, or minimizing the maximum eigenvalue of $(X^*X)^{-1}$. It turns out these three conditions are equivalent, since matrix (X^*X) has trace independent of the choice of angles (equal to mqR) and thus the minimum occurs when all eigenvalues are equal, and X^*X is R times the identity matrix. Generally speaking, the closer X^*X is to diagonal, the better.

Another optimality condition is to fix some vector w and minimize the variance $Var(w'\hat{c})$, which is a weighted sum of the entries of $\hat{c}$. This would be of interest the manufacturer when some harmonics, or some layers, are deemed to be more important than others.

1.8 Numerical Experiments

Random sampling of the m-dimensional hypercube $[0, 2\pi]^m$ with a given sample size R (number of tires) produces random vectors of angles $\Theta^1, \ldots, \Theta^R$, which are exponentiated to produce random matrices X. The block diagonal structrure of X^*X allows us to focus on one harmonic

at a time. Moreover, with the components of Θ^r chosen uniformly in $[0, 2\pi]$, then the mod 2π part of multiples $n\Theta^r$ are also uniformly distributed. Thus the Monte Carlo designs work for all harmonics.

Software code was produced in MATLAB to generate these random matrices and search for a best solution, typically from a sample of 10,000 to 100,000 random matrices. Sample code is provided in the appendix. Plots were obtained to show how the performance of MC-best solution improved with increasing R. A $1/R$ dependence was easily observed, although even for large values of R, the MC-best solution was not at the theoretical best solution, where X^*X is a multiple of the identity.

This led to the following theoretical observation. Recall for only one (n-th) harmonic, that

$$(X^*X)_{jk} = \sum_{r=1}^{R} e^{2\pi i n(\theta_k^r - \theta_j^r)}.$$

Thus on the diagonal, $(X^*X)_{jj} = R$, while on the off-diagonal, as $R \to \infty$,

$$\begin{aligned} \frac{1}{R}(X^*X)_{jk} &= \sum_{r=1}^{R} \frac{1}{R} e^{2\pi i n(\theta_k^r - \theta_j^r)} \\ &\to \int_0^1 e^{2\pi i n t} dt \\ &= 0, \end{aligned}$$

where the random sum is simply an approximation to the integral. Thus

$$\lim_{R \mapsto \infty} \frac{1}{R}(X^*X) = I_m,$$

which suggests that for large R one should see $(X^*X)^{-1}$ approximately equal to $\frac{1}{R} I_m$. Thus the Monte Carlo designs should be tending to this limit, although the convergence may be quite slow. In general, for uniform sampling, one expects

$$\sum_{r=1}^{R} \frac{1}{R} e^{2\pi i n(\theta_k^r - \theta_j^r)} \approx \frac{c}{\sqrt{R}}$$

for some constant $c > 0$.

A number of other numerical experiments were tried, including choosing random vectors of angles Θ where all but a few (say 4) of the angles were zero, and choosing angles from a discrete subset of $[0, 2\pi]$, say of 10 to 30 evenly distributed points. We tried to find interesting patterns in the resulting MC optimal designs, but did not see anything remarkable. However, these restricted MC designs have their use in practice, for instance when the operator constructing tires can only adjust a few layers at a time, or has only a limited precision in choice of angles. The $1/R$ behaviour was noted in these design too. MATLAB code for these experiments are included in the appendix.

1.9 The Prime Method

Having explored the the Monte Carlo method extensively, some ingenuity was required to find concrete patterns that would provide optimal methods without the MC search. An initial observation was that the matrix (X^*X) has off-diagonal terms which are sums of complex exponentials. It thus might be possible to arrange these exponentials to sum to zero. Indeed, note that

$$\sum_{r=1}^{R} e^{2\pi i(\frac{r}{R})} = 0,$$

hence if for each pair (j,k), the numbers $\{\theta_k^r - \theta_j^r\}_{r=1}^R$ are a permutation of the fractions $\{\frac{r}{R}\}_{r=1}^R$, then the matrix X is in its optimal form, with $(X^*X)^{-1} = \frac{1}{R} I_m$ exactly. This construction is then much better than the Monte Carlo method.

It turns out this can always be arranged when the number of layers m is a prime number, in which case the design size R is taken to be $R = m$. Here is an example of the matrix of angles for $m = 5$:

$$(\theta_j^r) = \frac{2\pi}{5} \cdot \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 2 & 3 & 4 \\ 0 & 2 & 4 & 1 & 3 \\ 0 & 3 & 1 & 4 & 2 \\ 0 & 4 & 3 & 2 & 1 \end{pmatrix} = \begin{pmatrix} \Theta^1 \\ \Theta^2 \\ \Theta^3 \\ \Theta^4 \\ \Theta^5 \end{pmatrix}.$$

Since $\mathbf{Z}/5$ is a field, it is easy to see that multiplication by n just permutes the rows of matrix (θ_j^k) (when $n \neq 0 \bmod 5$). That is, the matrix $(n\theta_j^k)$ also gives an optimal form, so this first design works equally for all harmonics which are not multiples of 5.

In general, for m equal to any prime, choose the design matrix of angles to be a $m \times m$ with entries

$$\theta_j^r = \frac{2\pi}{m}\{(j-1)(r-1) \bmod m\}.$$

Again, this design gives optimal X for any prime m and any harmonic n which is not a multiple of m. This is a result of the fact that $\mathbf{Z}/m$ is a field when m is prime, as shown in the following:

Theorem 1 *For integer m prime, n not a multiple of m,and design angles chosen as*

$$\theta_j^r = \frac{2\pi}{m}\{(j-1)(r-1) \bmod m\},$$

then the corresponding design matrix

$$X_{jk} = e^{n\theta_j^r}$$

is optimal.

Proof. The covariance matrix X^*X has entries $(X^*X)_{jk} = \sum_{r=1}^m e^{2\pi i n(\theta_k^r - \theta_j^r)}$ so along the diagonal, the exponentials are each equal to one, thus $(X^*X)_{jj} = m$. Off the diagonal, for any $j \neq k$, the exponentials have powers of the form $2\pi i(n(k-j)(r-1) \bmod m)$ for r in the range $1, 2, \ldots, m$. Since $n(k-j)$ is a non-zero element in the field $\mathbf{Z}/m$, multiplication by this element of the sequence $\{r-1\} = \{0, 1, \ldots, m-1\}$ simply permutes these elements of the field, so the

sum is over the m roots of unity. Hence each off-diagonal element of the matrix X^*X is zero. Thus X^*X is a multiple of the identity, and optimal. ■

In summary, this choice of design angles gives a powerful method both in that it gives explicitly an optimal solution, and that also works for a range of harmonics.

There are some disadvantages to this method. First it is somewhat inflexible in the number of layers m, as m must be prime. This can be remedied by introducing "fake" layers to reach the next lowest prime. For instance, for 20 layers, just pretend there are 23 layers, three of which are virtual. Or group two insignificant layers and call them one – so 20 becomes the prime 19.

Second, it requires using as many angles as there are layers. That is, if one has 19 effective layers, each of the 19 layers must be set to various angles as the tires are constructed. This can be an expensive, if not impossible, construction in some tire plants. A partial solution is to use a method of blocking layers, as discussed in the section below.

Finally, there is the problem of setting angles exactly: the operator may have only limited accuracy on how precisely layer angles can be set during construction, and may have physical obstruction in choosing particular angles. This method expects the operator to freely choose the angles.

1.10 The Blocking Method

To avoid the problem of setting many angles for many layers, it is convenient to block off groups of layers and treat them as a single unit. Indeed, this block layer can be rotated by zero degrees – in effect, no rotation – so only the remaining layers need to be rotated.

There is much flexibility in this method, as one can choose how many layers to move, how to group them, and so forth. Rather than explore all the possible permutations, here is a simple example with 12 layers, and setting no more than 5 angles at a time. One way to proceed is to group the first 8 layers as one, and treat the last four independently, giving 5 effective layers. Next, group the first and last four as one, middle four treated independently. Last, group the final eight as one.

Thus, the groupings of layers looks like the following:

actual:	1	2	3	4	5	6	7	8	9	10	11	12
group a:	1	1	1	1	1	1	1	1	2	3	4	5
group b:	1	1	1	1	2	3	4	5	1	1	1	1
group c:	2	3	4	5	1	1	1	1	1	1	1	1

The corresponding design of vectors of angles is given by choosing the 5×5 blocks of the last section, using the prime method. For clarity, we can show this array in block form, with blanks

indicating the blocked zeros, giving this 15×12 array:

$$\theta_k^r = \frac{2\pi}{5} \begin{pmatrix} & & & & & & & & 0 & 0 & 0 & 0 \\ & & & & & & & & 1 & 2 & 3 & 4 \\ & & & & & & & & 2 & 4 & 1 & 3 \\ & & & & & & & & 3 & 1 & 4 & 2 \\ & & & & & & & & 4 & 3 & 2 & 1 \\ & & & & 0 & 0 & 0 & 0 & & & & \\ & & & & 1 & 2 & 3 & 4 & & & & \\ & & & & 2 & 4 & 1 & 3 & & & & \\ & & & & 3 & 1 & 4 & 2 & & & & \\ & & & & 4 & 3 & 2 & 1 & & & & \\ 0 & 0 & 0 & 0 & & & & & & & & \\ 1 & 2 & 3 & 4 & & & & & & & & \\ 2 & 4 & 1 & 3 & & & & & & & & \\ 3 & 1 & 4 & 2 & & & & & & & & \\ 4 & 3 & 2 & 1 & & & & & & & & \end{pmatrix}.$$

In fact the zero rows are not particularly useful, since there are plenty of zeroes elsewhere in the matrix. Eliminating the zero rows give the following 12×12 matrix:

$$\theta_k^r = \frac{2\pi}{5} \begin{pmatrix} & & & & & & & & 1 & 2 & 3 & 4 \\ & & & & & & & & 2 & 4 & 1 & 3 \\ & & & & & & & & 3 & 1 & 4 & 2 \\ & & & & & & & & 4 & 3 & 2 & 1 \\ & & & & 1 & 2 & 3 & 4 & & & & \\ & & & & 2 & 4 & 1 & 3 & & & & \\ & & & & 3 & 1 & 4 & 2 & & & & \\ & & & & 4 & 3 & 2 & 1 & & & & \\ 1 & 2 & 3 & 4 & & & & & & & & \\ 2 & 4 & 1 & 3 & & & & & & & & \\ 3 & 1 & 4 & 2 & & & & & & & & \\ 4 & 3 & 2 & 1 & & & & & & & & \end{pmatrix}.$$

Exponentiating with $X_{kr} = e^{ni\theta_k^r}$, one finds the following covariance matrix with an elegant

block form:

$$(X^*X) = \begin{pmatrix} 12 & 7 & 7 & 7 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 \\ 7 & 12 & 7 & 7 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 \\ 7 & 7 & 12 & 7 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 \\ 7 & 7 & 7 & 12 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 & 12 & 7 & 7 & 7 & 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 & 7 & 12 & 7 & 7 & 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 & 7 & 7 & 12 & 7 & 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 & 7 & 7 & 7 & 12 & 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 12 & 7 & 7 & 7 \\ 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 7 & 12 & 7 & 7 \\ 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 7 & 7 & 12 & 7 \\ 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 7 & 7 & 7 & 12 \end{pmatrix}.$$

This is not an optimal matrix, but it is a reasonably good one whose performance is better than the MC designs found above. In general this method can be extended, using small primes to form $p \times p$ sub-design matrices which are used to tile the larger $R \times m$ full design matrix.

1.11 The GLP method

The Good Lattice Point (GLP) method of Fang and Wang (Ref. [**?**]) uses a careful choice of lattice points in an m-dimensional hypercube to accelerate integration over a multidimensional Riemann sum. The basic insight is to look for angle combinations which will lead to sequences $\{\theta_k^r - \theta_j^r\}_{r=1}^R$ which will allow for fast convergence of the sum

$$\sum_{r=1}^{R} \frac{1}{R} e^{ni(\theta_k^r - \theta_j^r)} \to \int_0^1 e^{2\pi i n x} dx.$$

With the Monte Carlo method, random sampling of the hypercube produces random sequences on $[0, 1]$, but with a slow convergence of order $R^{-\frac{1}{2}}$. The GLP method will exhibit convergence at the faster rate of $R^{-1} \log^m(R)$. We tested the GLP method to see if we would obtain a good sequence of angles, and came up with a surprising conjecture.

First, let us recap the definition of a lattice point set and a GLP set, as discussed in reference [**?**]. Let $(R, h_1, h_2, \ldots, h_m)$ be a vector of integers satisfying

- $m < R$
- $1 \leq h_j < R$
- $h_j \neq h_k$, for all $j \neq k$
- (h_j, R) are coprime, for all j.

The *lattice point set* of the generating vector $(R; h_1, h_2, \ldots, h_m)$ is the set of vectors $\{(x_{r1}, \cdots, x_{rm}), r = 1, \cdots, R\}$, with values

$$x_{rj} = \text{frac}\left(\frac{2rh_j - 1}{2R}\right),$$

where "frac" denotes the fractional part of the given real number. If this set has the smallest discrepancy (defined in Fang and Wang), it is called a GLP set.

The principle behind GLP sets is that generating vectors can always be constructed so that a GLP set is created, whose points are uniformly distributed about the hypercube. Fang and Wang tabulate many different choices for a range of R and m, corresponding in our case to numbers of tires R in the experimental design, and number of layers m per tire.

In our tire example, the vector of angles are obtained from the lattice point sets by scaling by a factor of 2π, so $\theta_j^r = 2\pi x_{rj}$. We tested a number of the GLPs from the book to see how close they are to optimal, and found in every instance, they were exactly optimal. We have the following:

Conjecture 1 *Every GLP set produces an optimal design: that is,*

$$(X^*X)^{-1} = \frac{1}{R} I_m \text{ exactly.}$$

Moreover,

- *m can be chosen arbitrarily (not necessarily prime)*
- *R can be chosen arbitrarity (although prime a popular choice)*
- *the same design is optimal for all harmonics co-prime with R.*

In the workshop, there was not enough time to explore how GLP sets were constructed in the literature, so it was not clear to us how optimal designs were resulting from these choices. A quick review of work in the area indicates some number-theoretical results are being used to construct the charts of Fang and Wang. However, a simple examination shows the differences $x_{rj} - x_{tk} = \text{frac}(\frac{r}{R}(h_j - h_k))$, so as in the prime method, the sum in the covariance matrix will cycle around a subset of the R roots of unity. For a good choice of the h_j, this subset will always sum to zero. Thus, while this is short of a proof verifying the GLP method works, there is the basis for a useful technique, explored in next section.

1.12 The Simplified Lattice Method

The GLP method is in fact too sophisticated for the problem at hand, as all that is needed are roots of unity that sum to zero. However, it suggests the following technique, which we call the Simplified Lattice Method.

Theorem 2 *Fix an integer $m > 0$ and fix $\mathbf{N}$ a subset of $\{1, 2, \ldots\}$. Suppose $(R; h_1, h_2, \ldots, h_m)$ is a vector of integers satisfying*

- $m < R$
- *R is not a divisor of $n(h_j - h_k)$ for all $n \in \mathbf{N}, j \neq k$.*

Then the vectors of angles (scaled lattice points) defined by

$$\theta_j^r = \frac{2\pi}{R}(rh_j \bmod R)$$

gives an optimal design for all harmonics $n \in \mathbf{N}$. That is,

$$(X^*X)^{-1} = \frac{1}{R} I_m.$$

Proof. For harmonic n in the set $\mathbf{N}$ and $j \neq k$, the integer $n(h_j - h_k)$ is not divisible by R and hence the map $r \mapsto n(h_j - h_k)r \bmod R$ defines a endomorphism on the ring $\mathbf{Z}/R$ which has more than one element in its range, a subring of $\mathbf{Z}/R$. Thus when scaled by $2\pi i$ and exponentiated, one obtains some R' roots of unity, for some divisor $R' > 1$ of R. Thus the terms in the sum

$$\sum_{r=1}^{R} e^{2\pi n r(h_k - h_j)/R}$$

simply cycle around these R' roots of unity, and so sum to zero. Hence the off-diagonal terms of the covariance matrix X^*X are zero, the diagonal terms are R, and the optimal design is achieved. ■

These criteria are easy to fulfill in any situation of tires, as shown in the following.

Example. With m the number of layers in the tire, and $\mathbf{N}$ a finite set of harmonics, let R be any prime number strictly bigger than m and all integers $n \in \mathbf{N}$. Then the integer vector $(R; 0, 1, 2, \ldots, m-1)$ generates lattice points yielding an optimal design.

This example gives a method much like the original prime method described above. However, the number of layers m need not be prime, and one can select any finite set of harmonics, yet still obtain an optimal design. The number of tires R need not be prime: one could choose a composite number with some prime factor bigger than m and all n.

There remains the disadvantage that almost every layer on almost every tire must be set to a non-zero angle, and the angles must be set to accuracies on the order of $2\pi/R$. Blocking may be used a a partial solution to this problem.

1.13 Robustness

We considered two ways in which a study may be corrupted:

- a tire is lost;
- there is some errors in the angle set.

A lost tire amounts to deleting a row from the matrix X. We considered a numerical example with $R = 23$, by removing a row at random, and measuring the eigenvalues of the resulting suboptimal matrix. The result was still close to optimal (i.e. eigenvalues nearly constant), even with two or three rows removed. Indeed, these suboptimal designs were still much better than our Monte Carlo searches. This was enough to convince the team that the optimal design was fairly robust; however we did not have time to investigate this more fully.

Errors in the angle setting can be investigated by a Monte Carlo study where normal random variates are added to the angles. Again, the design appears to be quite robust to such departures, although we note higher harmonics are proportionally more sensitive.

MATLAB code for both these investigations are included in the appendix.

1.14 Tire Types vs Replicates

Introducing tire replicates of the same type (that is, tires with the same vector of angles Θ^r) can help reduce the variance of the estimators and may be more cost-effective. This would be the case if it is cheaper to manufacture a run of several tires of the same type in a given study.

The linear model becomes

$$\vec{C}^{\vec{\Theta}} = (X \otimes 1_n)\vec{c} + \vec{\epsilon},$$

where $\otimes$ denotes the Kronecker matrix product, and n is the number of replicates of each type ot test tire. If one assumes the variance remains as $var(\epsilon) = \sigma^2 I_{nR}$, then

$$\begin{aligned} var(\hat{c}) &= ((X \otimes 1_n)^*(X \otimes 1_n))^{-1} (X \otimes 1_n)^* \sigma^2 (I_R \otimes I_n)(X \otimes 1_n) ((X \otimes 1_n)^*(X \otimes 1_n))^{-1} \\ &= ((X^*X \otimes 1_n^* 1_n))^{-1} \\ &= \frac{\sigma^2}{n}(X^*X)^{-1} \\ &= \frac{\sigma^2}{nR} I_R. \end{aligned}$$

Thus for good designs, the variance is inversely proportional to nR and it thus may be more effective to increase n and not R.

One must be careful to recognize that in the case of tire replicates, there are two different classes of error sources:

- errors specific to a particular type of tire (perhaps caused by inaccurate machine settings for the mold building one particular tire), and
- errors specific to individual tires or measurements.

The error structure is then of the form

$$var(\vec{\epsilon}) = \begin{pmatrix} \gamma^2 + \sigma & \gamma^2 & \gamma^2 & \dots & \gamma^2 \\ \gamma^2 & \gamma^2 + \sigma^2 & \gamma^2 & \dots & \gamma^2 \\ \vdots & & & & \vdots \\ \gamma^2 & \gamma^2 & \gamma^2 & \dots & \gamma^2 + \sigma^2 \end{pmatrix}_{n \times n} \oplus \dots \qquad \text{R times},$$

or more concisely, $var(\vec{\epsilon}) = \sigma^2 I_{nR} + \gamma^2 I_R \otimes (1_n 1_n^*)$, where σ^2 is a measure of the error in measurement, γ^2 the error in type. When both errors occur, repeating the calculation above shows the variance for the replicant test has the form

$$var(\hat{c}) = (\frac{\sigma^2}{n} + \gamma^2)(X^*X)^{-1}.$$

Thus while increasing n, the number of tires in each replicant set, will reduce the effect of the σ^2 errors, beyond a certain point it becomes necessary to reduce $(X^*X)^{-1}$ to affect the other error sources.

1.15 Summary

Over the course of the workshop, our team has developed some concrete solutions to the analysis problem in building tires with maximal uniformity. We have developed Monte Carlo methods to approximate optimal designs, found a number of general, explicit constructions for optimal designs, and demonstrated a blocking technique that addresses some of the complexity issues in the optimal designs that are relevant to the industrial practitioner. We have also investigated the robustness of the optimal designs, and examined the effects of replicating tests to improve performance of the analysis. MATLAB code for all these investigation have been included in the appendix of the report.

In addition, the techniques described would be useful in a variety of vibrational problems requiring the determination of the contributions to the harmonic components of a periodic signal.

1.16 Appendix

Over the course of the workshop, a number of short MATLAB scripts were produced to test some ideas, establish conjectures, run Monte Carlo methods, and generally explore ideas on the computer. Some of the more complete scripts are included here. Briefly, they are

- `stats.m` (Figure 1) Monte Carlo method for finding near optimal designs. A measure is provided of how the performance improves for increasing number of tires. Three optimality criteria are used.

- `stat1.m` (Figure 2) Another Monte Carlo method, with graphical output to demonstrate how the performance improves.

- `stat2.m` (Figure 3) Monte Carlo search for optimal designs, with only a limited number of layers adjusted at random – in this cases, only four layers may be moved for any one tire. Graphical output of performance.

- `minangle.m` (Figure 4) Chooses a near-optimal design using a Monte Carlo method, with limited precision on the angles. A display of the resulting angles is provided, to see if there are any useful patterns appearing in these random designs.

```
% file stats.m
% Monte Carlo method to find good designs, three different criteria
% See how perform improves as number of tires increase
zmean = [];
zstd = [];
for k = 20:5:80
   z = [];
   for j = 1:100,
      c = ones(20,1)/20;
      x = exp(2*pi*i*rand(k,20));
      y = inv(x'*x);
       z = [z; real(max(eig(y))), real(max(diag(y))), real(c'*y*c)];
    end
    zmean = [zmean;mean(z)];
    zstd = [zstd;std(z)];
end
zmean
zstd
```

Figure 1.1: Script stats.m

- `cyclic1.m` (Figure 5) Builds an optimal design using the prime method, and verifies that it is optimal.
- `Lattice.m` (Figure 6) Builds an optimal design using the GLP method, then randomly perturbs the design angles to simulate a technician with limited angle control. A test of robustness of the GLP optimal design.
- `removerow.m` (Figure 7) Beginning with an optimal design, randomly removes a row from the design matrix to see what happens to the performance of the design. Another test of robustness.

```
% File stat1.m
% Monte Carlo method to find good designs
% Some graphics output to help us see
zmean = [];
zstd = [];
zmin = [];
c = ones(20,1)/sqrt(20);
for k = 20:5:100
 z = [];
 for j = 1:100,
  x = exp(2*pi*i*rand(k,20));
  y = inv(x'*x);
   z = [z; real(max(eig(y))), real(max(diag(y))), real(c'*y*c)];
    end
  zmean = [zmean;mean(z)];
  zstd = [zstd;std(z)];
  zmin = [zmin;min(z)];
end
zmean
zstd
zmin
semilogy((20:5:100)' , zmean)
title('zmean')
pause
semilogy((20:5:100)' , zstd)
title('zstd')
pause
semilogy((20:5:100)' , zmin)
title('zmin')
```

Figure 1.2: Script stat1.m

```
% File stat2.m
% Monte Carlo method to find good designs, three different criteria
% Only four angles may be changed at a time
zmean = [];
zstd = [];
zmin = [];
c = ones(20,1)/sqrt(20);
for k = 25:5:100
   z = [];
   for j = 1:100
      x =[];
      for kk=1:k
         x = [x;exp(2*pi*i*rand(1,20).*(5>randperm(20)))];
      end
y = inv(x'*x);
   z = [z; real(max(eig(y))), real(max(diag(y))), real(c'*y*c)];
end
   zmean = [zmean;mean(z)];
   zstd = [zstd;std(z)];
   zmin = [zmin;min(z)];
end
zmean
zstd
semilogy((25:5:100)',zmean)
title('zmean, 4 angles')
pause
semilogy((25:5:100)',zstd)
title('zstd, 4 angles')
pause
semilogy((25:5:100)',zmin)
title('zmin, 4 angles')
```

Figure 1.3: Script stat2.m

```
%  File minangles.m
% This finds some good matrix, and the display what angles were chosen
CC = 5 ;  % the number of components (columns of X)
KK = 5 ;  % the number of tires (rows of X)
zmin = .1;  % the minimum value so far
zangles = [];% the angles at the minimum, so far
for j = 1:100000
  a = rand(KK,CC);
  x = exp(2*pi*i*a);
  y = inv(x'*x);
  z = real(max(eig(y)));
  if (z>zmin)
    j
    zmin = z
    zangles = a;
  end
end
'done - look at the plot!'
plot(zangles,'.')
title('angles by tire - max eigs')
pause
plot(zangles','.')
title('angles by component - max eigs')
```

Figure 1.4: Script minangles.m

```
%  File cyclic1.m  the cyclic (prime) method of choosing angles
%
%  p = a prime = number of components (including fakes) eg p = 19
%  h = which harmonic  1 <= h <= p-1
p = 19;
h = 1;
t = mod(h*(0:(p-1))'*(0:(p-1)) , p)/p; % cyclic choice of angles, scaled 0 to 1
x = exp(2*pi*i*t);
y = x'*x;
z = max(max(abs(y - p*eye(p))))
%  typically, z (the error) is size 10^{-15}
%  so we conclude x'*x = pI  (multiple of the identity matrix)
%  its inverse has eigenvalues 1/p
```

Figure 1.5: Script cyclic1.m

```
% File Lattice.m
% Computes a design via the GLP method, then introduces error in the angles
zmean = [];
zstd = [];
zmin = [];
c = ones(5,1)/sqrt(20);
h = [1,2,10,13,16];
l = 1:21;
a = (2*l'*h - 1)/(2*21);
t = a - floor(a);
%x = exp(2*pi*i*t)
%y = x'*x
%eig(y)
for k = 0:0.01:0.1
 z = [];
  for j = 1:100
   x = exp(2*pi*i*(t + k*randn(21,5)));
   y = inv(x'*x);
   z = [z; real(max(eig(y))), real(max(diag(y))), real(c'*y*c)]
  end
  zmean = [zmean;mean(z)];
  zstd = [zstd;std(z)];
  zmin = [zmin;min(z)];
end
zmean
zstd
zmin
plot((0:0.01:0.1) , zmean)
title('zmean')
pause
plot((0:0.01:0.1) , zstd)
title('zstd')
pause
plot((0:0.01:0.1) , zmin)
title('zmin')
```

Figure 1.6: Script Lattice.m

```
% File removerow.m
% This takes one of the nice matrices, removes a row at random
% and sees what happens to the eigenvalues
s = 5; % number of components
n = 21; % number of tires
h = [1,2,10,13,16];   % from the book
k = 1:n;
t = rem((2*k'*h-1)/(2*n),1);  % the angles, between 0 and 1
x = exp(2*pi*i*t);   % matrix
xp = x';
xq = xp(:,[1,2,4,5,6,7,10,11,12,13,15,16,17,18,19,20,21]);
y = xq*(xq');
   min(real(eig(y)))
```

Figure 1.7: Script removerow.m

Bibliography

[1] K.-T. Fang and Y. Wang, *Number-theoretic Methods in Statistics*, Chapman & Hall, London, 1994.

[2] T.W. Körner, *Fourier Analysis*, Cambridge University Press, Cambridge, 1988.

[3] R.C. St. John and N.R. Draper, *D-Optimality for Designs: A Review*, Technometrics, Vol. 17, No. 1, February 1975, pp. 15–22.