OPTIMAL EXCITATION SIGNAL DESIGN FOR FREQUENCY DOMAIN SYSTEM IDENTIFICATION USING SEMIDEFINITE PROGRAMMING

*Gergely Balázs Jávorszky** *Stephen Boyd*** *István Kollár** *Lieven Vandenberghe*** *Shao-Po Wu****

*Department of Measurement and Instrument Engineering, Technical University of Budapest,
Budapest, Műegyetem rkp. 9, Hungary, H-1521, Phone: + 36 1 463-1774, Fax: + 36 1 463-4112
Email: javor@mmt.bme.hu, kollar@mmt.bme.hu
**Information Systems Laboratory, Department of Electrical Engineering,
Stanford University, Stanford, CA 94305-4055, USA
Email: boyd@isl.stanford.edu, vandenbe@isl.stanford.edu, clive@isl.stanford.edu

ABSTRACT

The paper discusses two methods of optimal excitation signal design for identification with Maximum Likelihood parameter estimation: The "classical", dispersion function based method, and a new, semidefinite programming based one. It is shown that the dispersion function based algorithm is a primal-dual method. The problem can be formulated as matrix determinant maximization subject to linear matrix inequalities. We introduce an interior point method for excitation signal design. The implementations of the two methods are compared in practical use. For general problems, the semidefinite programming based approach performs better, while for practical optimal excitation signal design, the dispersion function based one is recommended.
Keywords: semidefinite programming, matrix inequalities, experiment design, optimal excitation signal design, determinant maximization, system identification.

1. INTRODUCTION

In frequency domain system identification, the usual goal is to identify a linear, time invariant system, that has a transfer function expressible in rational fraction form:

$$H(s) = e^{-sT} \frac{\sum_{k=1}^{nn} b_k s^k}{\sum_{k=1}^{nd} a_k s^k}$$

The parameters to be estimated can be arranged into a vector:

$$\mathbf{P} = [b_{nn},\ \ldots,\ b_1,\ a_{nd},\ \ldots,\ a_1,\ T]^T$$

We identify the system by exciting it with a periodic signal. The aim of excitation signal design is to distribute a given amount of signal power among the harmonic components so that the experiment is optimal in some sense.

Usually, the quality of the identified model is characterized by a scalar function of the Fisher information matrix $\mathbf{F}$ of the estimated parameters $\mathbf{P}$ [1, 2].

The commonly used method for optimal excitation signal design is based on the so-called dispersion function [1, 2, 3].

Semidefinite programming — or more generally, convex optimization — is well suited to the above optimal excitation signal design, since both the constraint set and the typical functions to be optimized are convex.

2. OPTIMAL EXCITATION SIGNAL DESIGN

We apply a multi-sine excitation signal at predefined radian frequencies $\omega_1,\ \omega_2,\ \ldots,\ \omega_F$:

$$x(t) = \sum_{k=1}^{F} \left(X_k \cos(\omega_k) + Y_k \sin(\omega_k) \right)$$

The frequencies $\omega_1,\ \omega_2,\ \ldots,\ \omega_F$ are taken from a sufficiently dense frequency grid.

We suppose that the delay parameter T is 0, or it has been measured in advance. The excitation signal (when applied to the system) and the measured output are distorted by noise. The noise is assumed to conform to a noise model, and some of the noise parameters are known [2], *eg.* they are determined from a priori measurements.

From the input and the output of the system, we can calculate the Maximum Likelihood estimate of the parameters by minimizing the so-called *cost function* with respect to $\mathbf{P}$. The cost function depends on the input, the parameters, and the known noise properties [2].

The Fisher information matrix of the parameters can be expressed in the following way:

$$\mathbf{F} = \sum_{k=1}^{F} \frac{X_k^2 + Y_k^2}{2} \mathbf{F}_k$$

where $\mathbf{F}_k$ is a symmetric, positive semidefinite partial information matrix belonging to frequency ω_k, and $\left(X_k^2 + Y_k^2\right)/2 = Q_k$ is the power carried by the excitation signal at ω_k. The total signal power $\sum_{k=1}^{F} Q_k$ is 1. The matrices $\mathbf{F}_k$ are of the form $\mathbf{F}_k = \mathbf{J}_k^T \mathbf{J}_k$. The cost function can be expressed as $C = \frac{1}{2}\mathbf{\Sigma}^T\mathbf{\Sigma}$, and from this, $\mathbf{J}_k = \partial[\mathrm{Re}(\epsilon_k),\ \mathrm{Im}(\epsilon_k)]^T/\partial\mathbf{P}$, where ϵ_k is the k^{th} element of Σ [1, 2, 3].

In excitation signal design, usually a scalar function of the Fisher information matrix is optimized. This scalar function is often the determinant of $\mathbf{F}$.

The inverse of the information matrix is the Cramér-Rao bound on the covariance matrix of the parameters estimated [1, 2]:

$$\mathrm{cov}(\mathbf{P}) \geq \mathbf{CR}(\mathbf{P}) = \mathbf{F}(\mathbf{P})^{-1}$$

The covariance of the estimated parameter vector $\mathbf{P}$ in most practical situations is close to the Cramér-Rao bound.

Therefore, an experiment that yields an information matrix that is "maximal" in some sense (*eg.* its determinant) can give a low variance parameter estimate.

2.1. Problem Statement with Matrix Inequalities

Let us assume that the total signal power is 1, and the partial information matrices $\mathbf{F}_1, \ \dots \ , \ \mathbf{F}_F$ (each symmetric, positive semidefinite with rank two) are given. We have to design an experiment $\mathbf{Q} = [Q_1, \ \dots \ , \ Q_F]$, that satisfies the following inequalities:

$$\mathbf{Q} \quad \geq \quad 0 \tag{1}$$

$$\mathbf{F}(\mathbf{Q}) = \sum_{k=1}^{F} Q_k \mathbf{F}_k \quad > \quad 0 \tag{2}$$

$$\sum_{k=1}^{F} Q_k \quad = \quad 1 \tag{3}$$

and $|\mathbf{F}(\mathbf{Q})|$ is maximal over the above constraint set. The inequality sign $>$ in (2) means that $\mathbf{F}(\mathbf{Q})$ is positive definite.

We can express the constraints in the following way: $\mathbf{g}(\mathbf{Q})$ is defined as

$$\mathbf{g}(\mathbf{Q}) = \left[\left(1 - \sum_{k=1}^{F} Q_k \right), \ Q_1, \ \dots \ , \ Q_F \right] \tag{4}$$

As long as $\mathbf{g}(\mathbf{Q})$ is non-negative, (1) is satisfied. The non-negativeness of (4) only yields that

$$\sum_{k=1}^{F} Q_k \leq 1$$

which does not equal to the constraint included in (3). To show that the conditions are however equivalent, let us suppose that $\mathbf{g}(\mathbf{Q}) \geq 0$ for a particular $\mathbf{Q}$; $\mathbf{F}(\mathbf{Q}) > 0$; $\sum_{k=1}^{F} Q_k = L < 1$ for a positive L. From this, it comes that $\frac{1}{L}\mathbf{Q}$ will satisfy (1) and (3); $\mathbf{g}\left(\frac{1}{L}\mathbf{Q}\right)$ will be non-negative; and

$$\left| \mathbf{F}\left(\frac{1}{L}\mathbf{Q}\right) \right| = \left(\frac{1}{L}\right)^{nn+nd+1} |\mathbf{F}(\mathbf{Q})| > |\mathbf{F}(\mathbf{Q})|$$

Therefore, if (1) and (3) are replaced with $\mathbf{g}(\mathbf{Q}) \geq 0$, at the optimum L will equal 1.

3. CONVEX OPTIMIZATION

The problem statement has reduced to the following form:

$$\begin{array}{lll} \text{maximize} & |\mathbf{F}(\mathbf{Q})| & \\ \text{subject to} & \mathbf{F}(\mathbf{Q}) \quad > \quad 0 & \\ & \mathbf{g}(\mathbf{Q}) \quad \geq \quad 0 & \end{array} \tag{5}$$

or equivalently, with the commonly used notations:

$$\begin{array}{lll} \text{minimize} & -\log|\mathbf{F}(\mathbf{Q})| & \\ \text{subject to} & \mathbf{F}(\mathbf{Q}) \quad > \quad 0 & \\ & \mathbf{g}(\mathbf{Q}) \quad \geq \quad 0 & \end{array} \tag{6}$$

The problem above is a matrix determinant maximization problem with linear matrix inequality constraints. Semidefinite programming — or more generally, convex optimization — deals with these problems [4].

3.1. The Dual Problem

Associated with (6) is the so–called *dual problem* [4]:

$$\begin{array}{lll} \text{maximize} & \log|\mathbf{W}| - \mathbf{g}_0^T \mathbf{z} + nn + nd + 1 & \\ \text{subject to} & \mathrm{tr}(\mathbf{F}_k \mathbf{W}) + \mathbf{g}_k^T \mathbf{z} = 0 & k = 1, \ \dots \ , \ F \\ & \mathbf{W} = \mathbf{W}^T > 0 & \\ & \mathbf{z} \geq 0 & \end{array} \tag{7}$$

where $\mathbf{W}$ (a matrix) and $\mathbf{z}$ (a vector) are the dual variables. The vectors $\mathbf{g}_k$ are $F+1$ element long, and defined in the following way:

$$\mathbf{g}_k = \mathbf{e}_{k+1} - \mathbf{e}_1; \quad k = 1, \ \dots \ , \ F \tag{8}$$

where $\mathbf{e}_k$ is the k^{th} unity vector and $\mathbf{g}_0 = \mathbf{e}_1$. Note: $\mathbf{g}(\mathbf{Q})$ can be expressed with $\mathbf{g}_0, \ \dots \ , \ \mathbf{g}_F$:

$$\mathbf{g}(\mathbf{Q}) = \mathbf{g}_0 + \sum_{k=1}^{F} Q_k \mathbf{g}_k$$

The *duality gap* is the difference between the primal and the dual objective:

$$\mathrm{tr}(\mathbf{F}(\mathbf{Q})\mathbf{W}) - \log|\mathbf{F}(\mathbf{Q})\mathbf{W}| - nn - nd - 1 + \mathbf{g}(\mathbf{Q})^T \mathbf{z} \tag{9}$$

The duality gap is always non-negative [4].

3.2. Primal-Dual Methods

The primal-dual methods optimize the primal objective. They calculate the dual variables, and then the duality gap. They stop as soon as the duality gap has become less than an "absolute tolerance" (that may be a parameter to the algorithm). This feature provides us with a "certificate" of the optimality of the results conveyed, since the optimum is in between the primal and dual solutions.

Note: The primal-dual methods solve (6), therefore the absolute tolerance is related to $-\log|\mathbf{F}(\mathbf{Q})|$. In optimal excitation signal design (5) is to be solved. If the absolute tolerance reached was ε, then

$$1 \leq \frac{|\mathbf{F}(\mathbf{Q}_{result})|}{|\mathbf{F}(\mathbf{Q}_{opt})|} \leq e^{\varepsilon}$$

holds, where $\mathbf{Q}_{opt}$ is the vector that optimizes (6), and $\mathbf{Q}_{result}$ is the result of calculations.

4. METHODS OF DETERMINANT MAXIMIZATION

In this section, two algorithms for solving (6) (or (5)) will be discussed. One of them is known, the other one is presented here. We will illustrate the differences between the two and give suggestions when each one should be used.

4.1. Determinant Maximization using the Dispersion Function

The dispersion function is defined as [2, 3]

$$\nu(\omega_k) = \mathrm{tr}\left(\mathbf{F}(\mathbf{Q})^{-1} \mathbf{F}_k\right) \tag{10}$$

provided that the total signal power is 1.

We can construct a simple iterative algorithm based on the dispersion function [2, 3]. As the first step, an input

signal is constructed by distributing the total signal power evenly among the frequencies ω_1, ω_2, ... , ω_F. In each iteration, the dispersion of the current input signal is calculated for every frequency ω_k, and the next approximation $\mathbf{Q}_{i+1}$ of the optimal excitation signal amplitudes will be

$$Q_{k,i+1} = Q_{k,i} \frac{\nu(\omega_k)}{nn + nd + 1}$$

where $Q_{k,i}$ is the k^{th} element of $\mathbf{Q}_i$.

The excitation signal amplitudes are always sum up to 1 (in other words, the first element of $\mathbf{g}(\mathbf{Q})$ is always 0). This algorithm strictly monotonically converges to the optimum [1, 2].

The algorithm can be stopped as soon as

$$\max_k (\nu(\omega_k)) - nn - nd - 1 \tag{11}$$

becomes less than an "absolute tolerance". Expression (11) reduces in each iteration step.

From this point on, we refer to the dispersion function based iterative algorithm as *DF*.

4.2. DF and Duality

DF is a primal-dual method, with (11) being the duality gap. To prove this, first we need to construct a dual $\mathbf{W}$ and $\mathbf{z}$ for the current iterate $\mathbf{Q}$. Let $\mathbf{W}$ be $\mathbf{F}(\mathbf{Q})^{-1}$. Then, from (7) and (10),

$$\mathrm{tr}\left(\mathbf{F}_k \mathbf{F}(\mathbf{Q})^{-1}\right) = \nu(\omega_k) = -\mathbf{g}_k^T \mathbf{z}$$

for $k = 1, \ldots, F$. From (8), it follows that $\mathbf{g}_k^T \mathbf{z} = z_{k+1} - z_1$, that is

$$z_{k+1} = z_1 - \nu(\omega_k) = z_1 - \mathrm{tr}\left(\mathbf{F}_k \mathbf{F}(\mathbf{Q})^{-1}\right)$$

where z_i is the i^{th} element of $\mathbf{z}$. Now, if we assign

$$z_1 = \max_k (\nu(\omega_k))$$

we get a dual $\mathbf{W}$ and $\mathbf{z}$, that satisfies the constraints in (7). The duality gap (9) with these constructed $\mathbf{W}$ and $\mathbf{z}$ will be

$$\begin{aligned} & \mathrm{tr}\left(\mathbf{F}(\mathbf{Q})\mathbf{F}(\mathbf{Q})^{-1}\right) - \log\left|\mathbf{F}(\mathbf{Q})\mathbf{F}(\mathbf{Q})^{-1}\right| \\ & -nn - nd - 1 + \mathbf{g}(\mathbf{Q})^T \mathbf{z} \\ = \quad & nn + nd + 1 - 0 - nn - nd - 1 + \mathbf{g}(\mathbf{Q})^T \mathbf{z} \end{aligned}$$

from (4), it further equals to

$$\begin{aligned} & \sum_{k=1}^{F} z_{k+1} Q_k \\ = \quad & \sum_{k=1}^{F} Q_k \left(z_1 - \mathrm{tr}\left(\mathbf{F}_k \mathbf{F}(\mathbf{Q})^{-1}\right)\right) \\ = \quad & \sum_{k=1}^{F} Q_k z_1 - \sum_{k=1}^{F} \mathrm{tr}\left(\mathbf{F}_k Q_k \mathbf{F}(\mathbf{Q})^{-1}\right) \\ = \quad & z_1 - \mathrm{tr}\left(\mathbf{F}(\mathbf{Q})\mathbf{F}(\mathbf{Q})^{-1}\right) \\ = \quad & \max_k (\nu(\omega_k)) - nn - nd - 1 \end{aligned}$$

which is the same as (11).

4.3. Properties of DF

Because the total signal power in each iteration is 1 (*ie.* the first element of $\mathbf{g}(\mathbf{Q})$ is always 0), DF is not an *interior point method*, unlike the one described later in section 4.4.

The theoretical convergence of DF has not yet been covered. Since it is a gradient-like method [1], and not a Newton or quasi-Newton one, its convergence is most probably *worse* than that of the method introduced in section 4.4., at least in terms of complexity (number of iteration steps). However, the exact statement needs further research.

4.4. An Interior Point Method

Recently, a more effective interior point primal-dual method has been developed for determinant maximization. The detailed description of the algorithm can be found in [4]. This algorithm will be referred to as *MAXDET* in this paper.

MAXDET requires for a worst case run

$$\mathcal{O}\left(\sqrt{(1+F)}\log\left(\varepsilon^{(0)}/\varepsilon\right)\right)$$

number of Newton iterations, where $\varepsilon^{(0)}$ is the initial duality gap, and ε is the desired absolute tolerance [4]. The term $\log\left(\varepsilon^{(0)}/\varepsilon\right)$ can be regarded as a constant value, since it does not grow fast with reducing ε and, in most practical situations, the achievable values of ε are limited below by the computing platform accuracy and the roundoff noise of calculations.

The complexity of one Newton step depends heavily on the problem structure. Generally, the complexity is $\mathcal{O}\left(F^2\left((F+1)^2 + (nn+nd+1)^2\right)\right)$.

Numerical experiments indicate that behavior is much better in practice than the worst case. The number of iterations usually lies between 5 and 50, almost independently of problem dimensions [4].

5. IMPLEMENTATIONS OF *DF* AND *MAXDET*

In this section, the implementations of the two algorithms are compared. Testing in a practical setting is very important. A theoretically "good" implementation can perform much worse in practice than a "bad" one if its goodness only shows up with extremely large problem sizes.

The theoretical *memory complexity* of the programs is in the range of the size of the matrices involved, independently of the absolute tolerance required. Thus, we only measured the *time complexity* versus the absolute tolerance.

5.1. Comparability of Implementations

Both algorithms are primal-dual methods (see sections 4.1. and 4.4.). They stop as soon as they have reached a sufficiently small duality gap. In this sense, DF and MAXDET are *theoretically* comparable.

However, implementation and algorithm always differ. We prepared a list of aspects of implementation comparability. In the testing, we paid particular care to the points below:

- The two programs have to start their iterations from the same $\mathbf{Q}$.
- Convergence of MAXDET depends on a parameter γ [4]. We had to find the appropriate value of γ for excitation signal design.

- DF has an optimized Matlab implementation [5], while that of MAXDET is only a pilot version.
- To the contrary, MAXDET has an optimized binary executable (coded in C [4]), but DF does not.
- The implementation of DF [5] has to be slightly modified because it does not take into account the duality gap in the stopping criterion. DF only exits if the maximum number of iterations (a parameter to the program) has been exceeded.
- The implementation of MAXDET [4] also has to be modified. Although it exits if the duality gap has become less than the absolute tolerance but it also exits if the "relative tolerance" has been reached or the "maximum number of Newton iterations" has been exceeded (both are parameters to the program).
- For the comparison, we needed to run the programs on the same platform.
- Other circumstances (*eg.* free memory, processor time etc.) also have to be the "same".

In the following sections, the points above will be addressed.

5.2. Input to the Implementations

DF starts iterating from the vector (see section 4.1.)

$$\mathbf{Q}_{start} = \left[\frac{1}{F}, \ \cdots \ , \ \frac{1}{F}\right]$$

($\sum_{k=1}^{F} Q_k = 1$, since DF *is not* an interior point algorithm). MAXDET only works with a particular $\mathbf{Q}$, if that satisfies the constraints of (6) and $\mathbf{g}(\mathbf{Q}) > 0$ (*ie.* $\sum_{k=1}^{F} Q_k < 1$, because MAXDET *is* an interior point algorithm). In other words, MAXDET *cannot start from* $\mathbf{Q}_{start}$. We overcame the problem by providing MAXDET $\frac{9}{10}\mathbf{Q}_{start}$. Convergence of MAXDET does not depend heavily on the starting $\mathbf{Q}$.

We run MAXDET with various values of the parameter γ. The range for γ recommended in [4] was 10–1000. We tested MAXDET for every problem and for every tolerance with γ values chosen evenly (with the difference of 50) from this range. MAXDET did not show dramatic differences in behavior to this parameter. The greatest difference in runtime was less than twofold. The optimal value of γ depended upon the tolerance. We concluded that γ — at least in excitation signal design — would not affect seriously the performance.

In its original form, MAXDET exits under 3 conditions [4], namely either

- Absolute tolerance is reached
- "Relative tolerance" is reached
- Maximum number of Newton iterations is exceeded.

We had developed a "shell procedure" for MAXDET. This procedure runs MAXDET in a loop as long as it does not reach the absolute tolerance. If MAXDET exits under the second condition, the "shell procedure" decreases the value of relative tolerance parameter for the next run. Similarly, if the third condition was true on exit, the maximum iteration number parameter is increased. Note: In each iteration of the "shell procedure", MAXDET starts the iterations from $\frac{9}{10}\mathbf{Q}_{start}$.

The original implementation of DF exits if the maximum number of iterations has been exceeded. However, it calculates but ignores the duality gap in each iteration. We had carried out a straightforward modification of the implementation to make it a "pure" primal-dual program.

5.3. Platform and Compilation

A *PC (486 DX2 at 66MHz, 16Mb RAM) with Windows 3.1 operating system* was chosen as a platform. The most important consideration of this selection was that this platform provided more control over program running than other platforms, because

- A process does not have to compete with other processes for the processor, since Windows 3.1 is a single user, single task environment.
- There are no operating system "daemon" processes, or they can be killed in advance. And in general, the user can check and determine which processes are running in a given time instance.
- Matlab had been implemented under Windows.

In this way, we could provide that both programs had the same processor power as the other, and as themselves had had in the previous runs.

We had compiled DF to C using the Matlab to C compiler [6]. Before compilation, we had removed the convenience features (*eg.* graphical output, convergence statistics) and unnecessary calculations (*eg.* the calculation of the determinants and the Cramér-Rao bound) from the Matlab code. Also, preallocated matrix space and the so-called *type imputation* [6] had been employed to further speed up the program. The binary executable had been produced using the Watcom C optimizing compiler.

MAXDET had been efficiently implemented in C [4]. We had compiled it to the binary format using the *same* compiler with the *same* compiler options as for DF, providing a comparable setting for testing.

The Matlab "shell procedure" that we had developed for MAXDET had been compiled in the same way as DF, and then it had been linked with the MAXDET executable.

5.4. Matlab versus Binary Executable Implementations

We had decided to compare both the Matlab and the binary executable implementations. As it has been pointed out above, DF has not had a binary executable implementation, and MAXDET has not had an optimized Matlab implementation. Thus, we decided to declare the comparison *valid* if the relative behavior of the Matlab implementations and the binary executable implementations are similar. The Matlab implementations were tested on the PC. We also tested the Matlab files on a *Sun workstation* to further improve the validity of comparison.

Note: The term "binary executable" refers to the Matlab *MEX* files. The reason behind choosing Matlab as the outer testing environment was that Matlab provided a consistent and platform independent interface over the actual hardware.

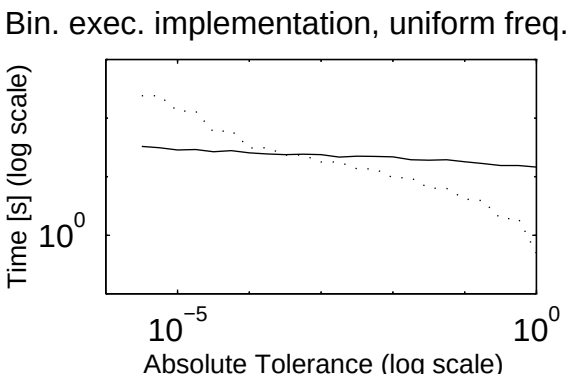

Figure 1. Results of runtime analysis on a PC with a sample system. *Legend:* solid line: MAXDET, dotted line: DF.

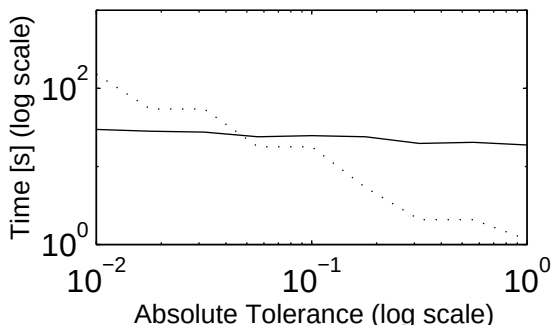

Figure 2. Results of runtime analysis on a PC with a sample system. *Legend:* solid line: MAXDET, dotted line: DF.

6. COMPARISON OF THE IMPLEMENTATIONS

We have tested the two programs using various inputs. Runtime results of two experiments with the binary executable versions can be seen in figures 1 and 2. The experiment results with the Matlab versions are in figures 3 and 4.

The system identified can be found in [2], section 4.3.5, page 179.

The frequency vector $[\omega_1, \ \ldots \ , \ \omega_F] \, / \, (2\pi)$ used for figures 1 and 3 is [20 Hz, 40 Hz, 60 Hz, ... , 1000 Hz] — *ie.*, the frequencies are uniformly distributed in a wide band.

In figures 2 and 4, the frequency grid of the spectrum has five narrow bands: 340–360 Hz, 390–410 Hz, 490–520 Hz, 620–640 Hz, and 700–720 Hz. The difference between frequency points in each band is 2 Hz. This distribution of frequency points results a more complex problem structure.

Note: the figures verify the fact that the Matlab implementation of DF is highly optimized even in comparison to the binary executable implementations. The Matlab version of DF exploits all the optimized features of the Matlab interpreter, so compilation does not results tremendous changes in runtime.

6.1. Discussion of the Results

The tolerance range used in the experiments was "typical" with respect to usual problems in practice.

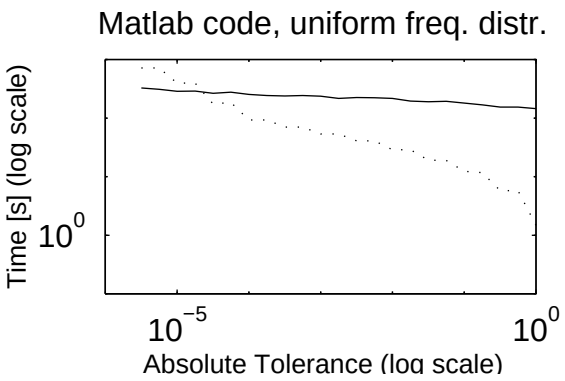

Figure 3. Results of runtime analysis with a sample system. The relative behavior of the two programs is similar to that of figure 1, although the curves cross elsewhere. *Legend:* solid line: MAXDET, dotted line: DF.

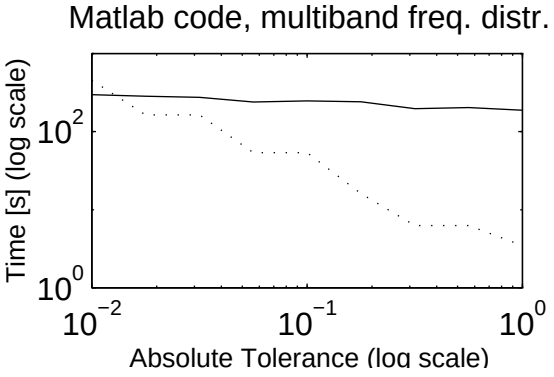

Figure 4. Results of runtime analysis with a sample system. The relative behavior of the two programs is similar to that of figure 2, although the curves cross elsewhere. *Legend:* solid line: MAXDET, dotted line: DF.

After various experiments, we found the relative behavior of Matlab implementations of MAXDET and DF similar to the relative behavior of the binary executable implementations. Therefore, it is probable that the algorithms behave in a similar way relative to each other in any implementation in the tolerance range inspected.

Typically, the curve of DF crosses the curve of MAXDET and reveals an exponential behavior of DF to the tolerance parameter. In some runs, MAXDET runtime was negligible compared to that of DF (*ie.* DF was slower by orders of magnitude), especially when small but sensible tolerance had to be achieved.

On the contrary, runtime of MAXDET could have been regarded as almost "constant", in a wide tolerance range.

Both implementations turned out to be sensitive to the problem structure. Again, MAXDET behaved relatively better.

6.2. Conclusions for Excitation Signal Design

It can be concluded from the experiments that MAXDET is generally faster, except for low tolerances and simple problems.

However, for excitation signal design, very low tolerances are satisfactory [2, 3]. At low tolerances, DF is considerably (up to orders of magnitude) faster than MAXDET. Thus, DF is still useful for engineering purposes.

6.3. Future Research

Some areas that have not yet been covered are the following:

- Theoretical results on the convergence of DF.
- MAXDET is a general program for determinant maximization. A version that is specifically developed for optimal excitation signal design may perform much better.
- From the figures, it seems that MAXDET needs a lot of iterations to achieve a low tolerance, but with only a few number of additional iterations a high tolerance range is covered. DF needs much less time (and iterations) for low tolerances. The two algorithms might be combined in such a way, that the initial $\mathbf{Q}$ estimates are conveyed by DF, then MAXDET refines the solution to a very high tolerance. The time needed for this combined algorithm may be much less than that of either one.

Future research will also include other approaches to optimal excitation signal design (*eg.* design of signals with prescribed crest factor [2, 3]).

REFERENCES

[1] Goodwin, G. and R. L. Payne, Dynamic System Identification: Experiment Design and Data Analysis. Academic Press, New York, 1977.

[2] Schoukens, J. and R. Pintelon, Identification of Linear Systems: A Practical Guideline for Accurate Modeling. Pergamon Press, Oxford, UK, 1991.

[3] Schoukens, J., P. Guillaume and R. Pintelon, "Design of Broadband Excitation Signals," in: "Perturbation Signals for System Identification," edited by K. Godfrey. Prentice Hall, Englewood Cliffs, 1993.

[4] Vandenberghe, L., S. Boyd and S-P. Wu, Determinant maximization with linear matrix inequality constraints. Report, Information Systems Lab., Stanford University, 1996.

[5] Kollár, I., Frequency Domain System Identification Toolbox for Matlab. The MathWorks, Natick, MA, 1994.

[6] The Matlab Compiler User's Guide. The MathWorks, Natick, MA, 1995.