Matrix functions, correlation matrices, and news from NAG

Craig Lucas

June 2011

nag® Experts in numerical algorithms and HPC services

Contents

- Matrix Functions
- Nearest Correlation Matrices
  - Improvements to our first routine
  - Weights and forcing positive definiteness
  - Factor Model
- Recent News and Coming Soon from NAG

NAG Products

- Libraries & Toolbox:
  - NAG Fortran Library, NAG C Library
  - NAG Library for SMP & Multi-core
  - NAG Toolbox for MATLAB, NAG Library for .NET
- Callable from many environments
  - C++, Excel, F#, Java, Python, R,…
- Library code written and contributed by some of the world's most renowned mathematicians and computer scientists
- Algorithmic development in house and in collaboration with academia and industry
- NAG Fortran Compiler and Fortran Builder
  - The World's Best Checking Compiler… essential for building a Library!

NAG Library and Toolbox Contents

- Root Finding
- Summation of Series
- Quadrature
- Ordinary Differential Equations
- Partial Differential Equations
- Numerical Differentiation
- Integral Equations
- Mesh Generation
- Interpolation
- Curve and Surface Fitting
- Optimization
- Approximations of Special Functions
- Dense Linear Algebra
- Sparse Linear Algebra
- Correlation & Regression Analysis
- Multivariate Methods
- Analysis of Variance
- Random Number Generators
- Univariate Estimation
- Nonparametric Statistics
- Smoothing in Statistics
- Contingency Table Analysis
- Survival Analysis
- Time Series Analysis
- Operations Research

MATRIX FUNCTIONS

Matrix Functions

- Matrix functions examples:
  - A inverse
  - exp(A)
  - sqrt(A)
- NAG working with Prof Nick Higham under a KTP Partnership. Associate Edvin Deadman.
- Developing a suite of codes, and making improvements to state of the art.
- Nick Higham also has €2M EU grant for more algorithm development.

Matrix Functions - Definition

- Several definitions, one is:
- Suppose the function *f* has Taylor series expansion:

$$f(z) = \sum_{k=0}^{\infty} a_k (z-\alpha)^k, \quad a_{k=} \frac{f^{(k)}(\alpha)}{k!}$$

- with radius of convergence *r*, then if max$|\lambda_i - \alpha| < r$ we have:

$$f(A) = \sum_{k=0}^{\infty} a_k (A - \alpha I)^k$$

- *f* could be sin, exp, ... or user defined.

Matrix Functions Examples

- Solution of differential equations:
  - exponential
  - trigonometric functions
- Finance – transition matrices describe evolution from one time-step to the next:
  - p^{th} roots
  - logarithm
  - exponential
- Population dynamics, NMR spectroscopy, optics, graphs and maps …

Matrix Functions Examples

- Vectors $\boldsymbol{v}_{2011}$ and $\boldsymbol{v}_{2010}$ represent credit states or stock prices in 2011 and 2010.

$$\boldsymbol{v}_{2011} = \boldsymbol{P}\,\boldsymbol{v}_{2010}$$

- $\boldsymbol{P}$ is the *transition matrix*.
- Using $\boldsymbol{P}^{1/2}$ enables predictions to be made at 6-monthly intervals
- *Transition intensity matrix* $\boldsymbol{Q}$ satisfies

$$\boldsymbol{P}(t) = e^{\boldsymbol{Q}t}$$

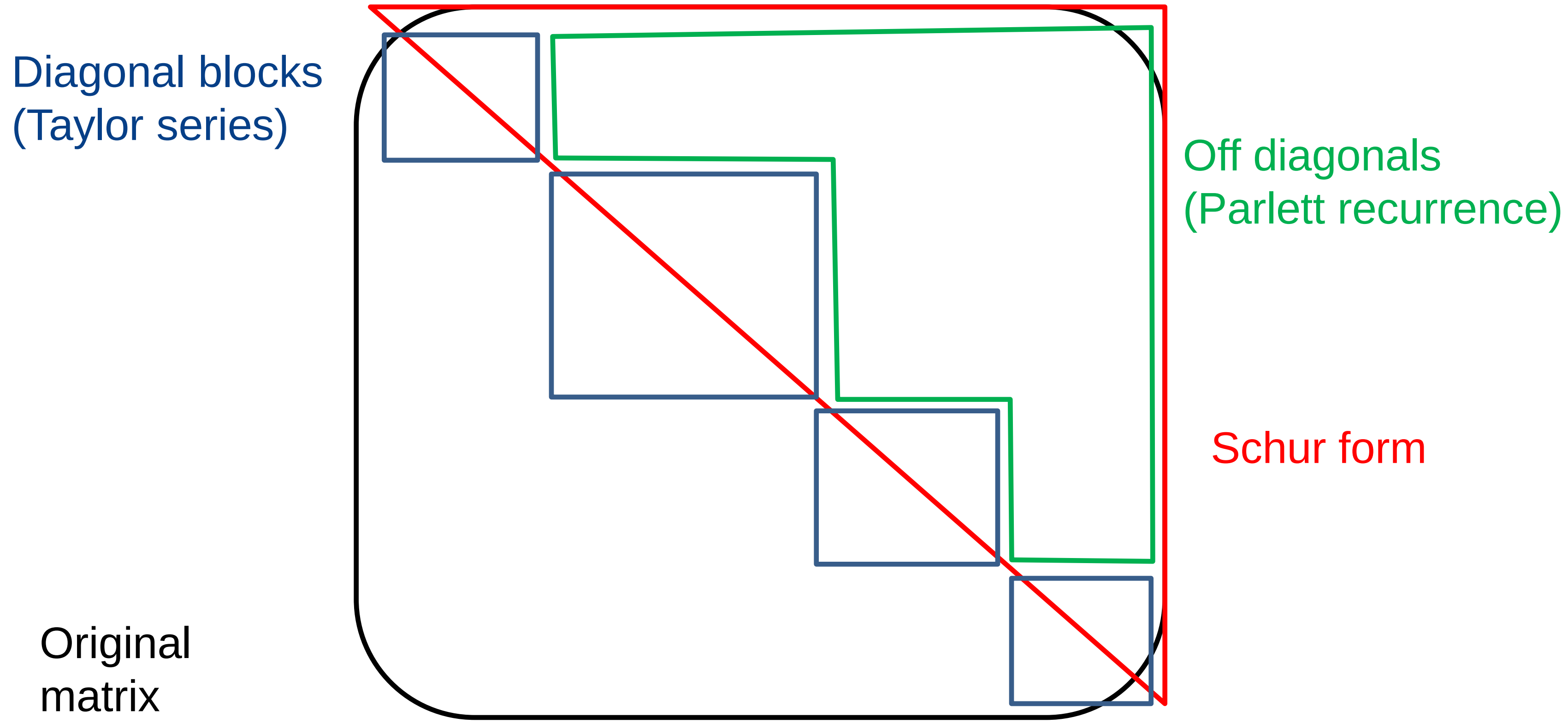
The Schur-Parlett Algorithm

- New routines in the NAG Library based on Schur-Parlett Algorithm.
- General purpose algorithm for computing a function of a complex matrix, *f(A)* (Davies & Higham, 2003)
- Using for Matrix sine, cosine, exponential, sinh, cosh
- And an adapted algorithm for matrix logarithm

The Schur-Parlett Algorithm

1. Convert into Schur form.
   - If a real matrix has complex eigenvalues then we have to work in complex arithmetic.
2. Reorder the Schur decomposition to group eigenvalues of similar size into blocks, T_{ii}
3. Compute $f(T_{ii})$ via Taylor series
4. Compute off-diagonal blocks via a Parlett recurrence
5. Transform back to original form

The Schur-Parlett Algorithm



The Schur-Parlett Algorithm

- As we use a Taylor series we need an arbitrary number of derivatives, $f^{(K)}(x)$ for all k and x.
- User supplied functions, therefore, require numerical differentiation. (Unless passed by user.)
- Implemented Lyness & Moler (1967) which is based on Cauchy's theorem.
- Importantly an error estimate is returned.
- Will also be a new user callable routine.

Matrix Functions in the NAG Library

- Current routines :
  - Matrix exponential - scaling and squaring
  - Symmetric/Hermitian exponential and general matrix function.

- Coming soon:
  - Real and complex matrix sine, cosine, exponential, sinh, cosh
  - Real and complex matrix logarithm
  - Function of a general matrix, with or without user supplied derivatives

Matrix Functions in the NAG Library

- Coming over the next 1-3 years.
  - norm / condition number estimation
  - square root
  - p^{th} root
  - **‘action of exponential’ on a vector, exp(A)v**
  - other algorithms for exp, sin and cos

- You set the priorities!

NEAREST CORRELATION MATRICES

Correlation Matrices

- An *n*-by-*n* matrix is a *Correlation Matrix* if
  - it is symmetric
  - has ones on the diagonal
  - its eigenvalues are nonnegative (positive semidefinite)

- The off diagonal entries are bounded by -1 and 1

Correlation Matrices

- Empirical correlation matrices are often not mathematically true due to inconsistent or missing data.
- Thus we are required to find the *nearest correlation* matrix.
- For example, we need to solve the convex optimization problem:

$$\min \tfrac{1}{2}\left\|G - X\right\|_F^2$$

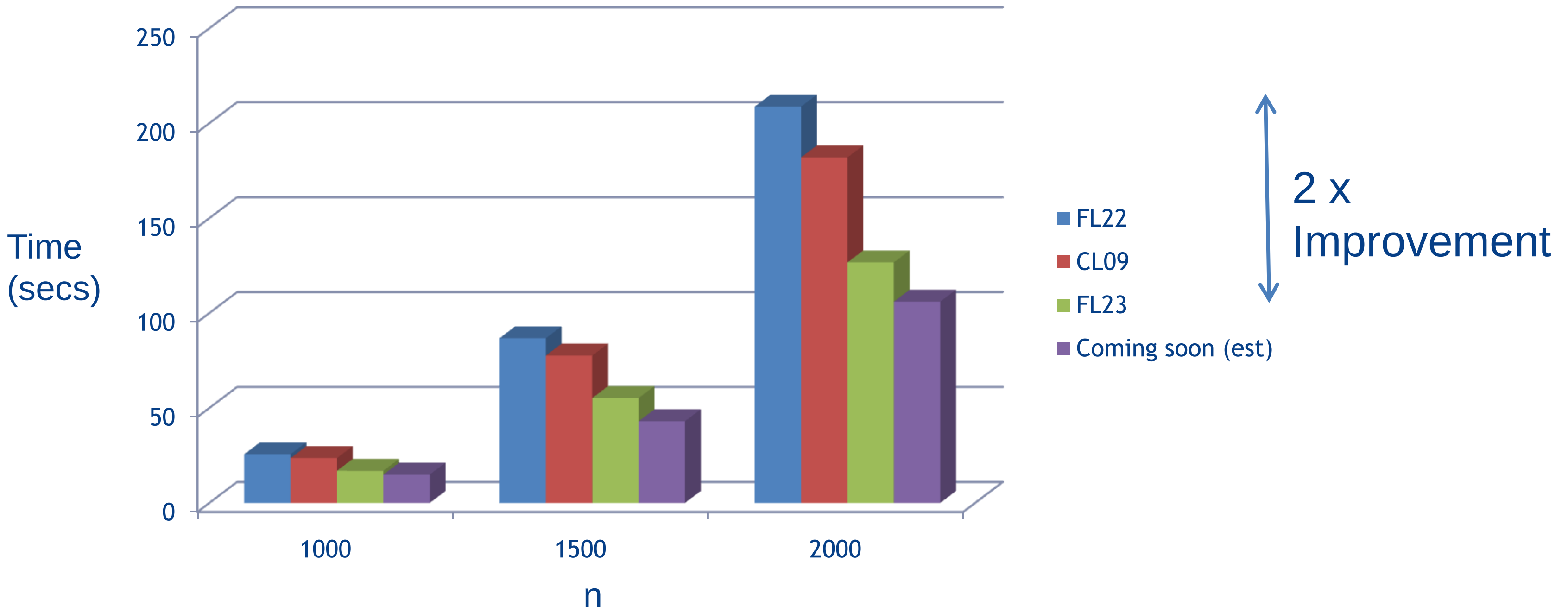
- to find *X*, a *true* correlation matrix, where *G* is an approximate correlation matrix.

Nearest Correlation Matrices

- G02AA applies an inexact Newton method to a dual (unconstrained) formulation of this problem.
- Algorithm by Qi and Sun (2006). It applies the improvements suggested by Rüdiger Borsdorf* and Higham (2010). *NAG funded PhD.
- It is globally and quadratic convergent.
- We have made some improvements ...

Nearest Correlation Matrices

- Code and algorithmic improvements



- Improvements depend on data properties

Weights and Eigenvalues

- New routine, G02AB, extending G02AA, to introduction weights and forcing of the computed NCM to be positive definite.
- Finds the nearest correlation matrix *X* by minimizing the following, where *G* is an approximate correlation matrix, using the *weighted* Frobenius norm:

$$\frac{1}{2}\left\|W^{\frac{1}{2}}(G - X)W^{\frac{1}{2}}\right\|_F^2$$

- where *W* is diagonal.

Weights and Eigenvalues

- The effect of W:

```
A =
     0.4218     0.6557     0.6787     0.6555
     0.9157     0.3571     0.7577     0.1712
     0.7922     0.8491     0.7431     0.7060
     0.9595     0.9340     0.3922     0.0318
W = diag([10,10,1,1])


W*A*W =
    42.1761    65.5741     6.7874     6.5548
    91.5736    35.7123     7.5774     1.7119
     7.9221     8.4913     0.7431     0.7060
     9.5949     9.3399     0.3922     0.0318
```

Whole rows/cols weighted by w_i

Elements weighted by w_i * w_j

Weights and Eigenvalues

- Note that if the weights vary by several orders of magnitude from one another the algorithm may fail to converge.

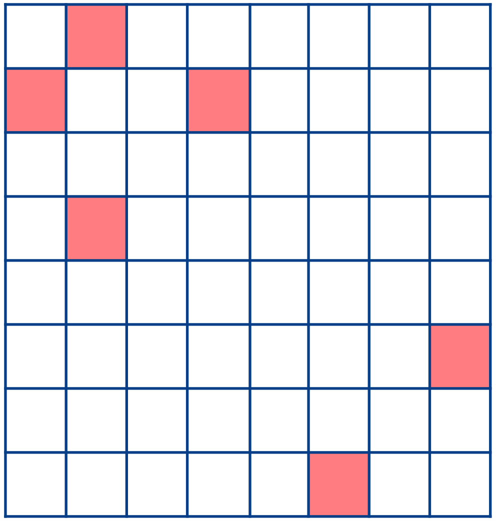
- You can optionally specify a lower bound on the eigenvalues of the computed correlation matrix
- Forcing the matrix to be positive definite, with minimum eigenvalue $0 < \alpha < 1$.

Other weighting possibilities

- Suppose some elements are known to be true.
- Hardamard weighting to fix elements

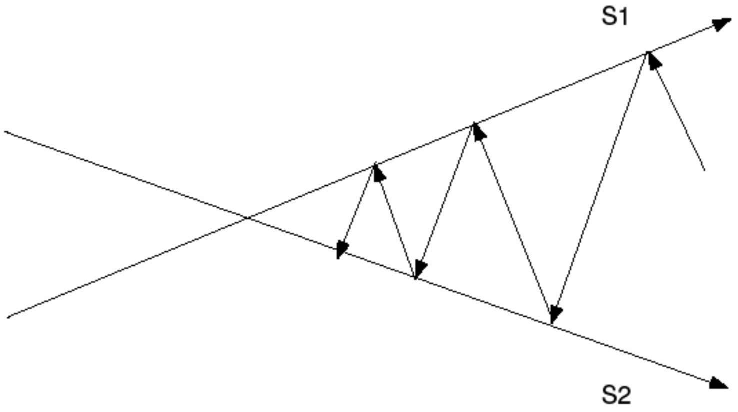
$$\min \tfrac{1}{2}\left\| H \circ (G - X) \right\|_F^2$$

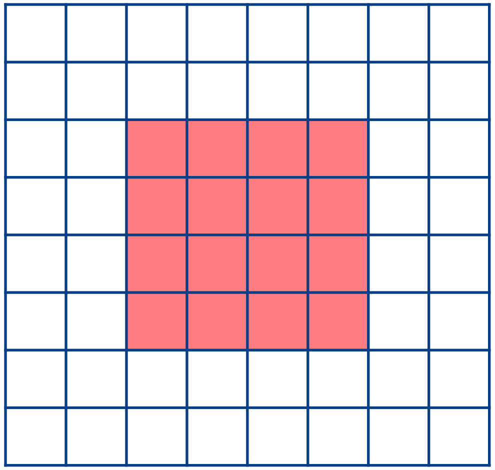
- Element-wise multiplication by *H*.
- Planned for next release.



Other weighting possibilities

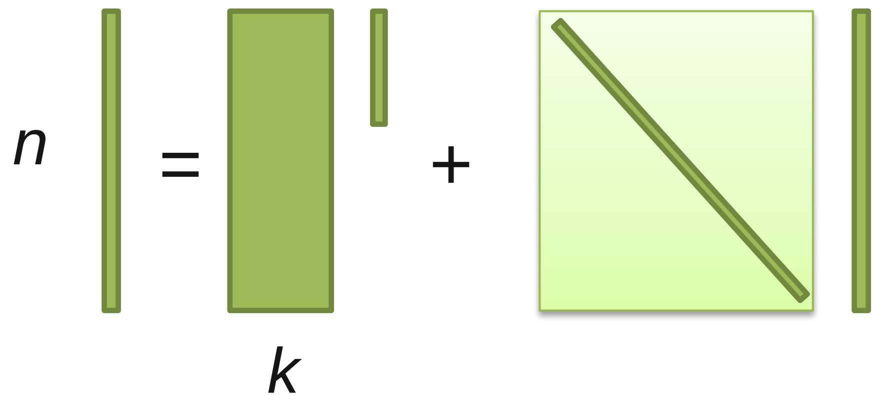
- "Alternating projections" algorithm.
- Project onto convex sets: semidefinite and unit diagonal.
- Can fix a whole sub-matrix with third projection.
- Linear convergence.





Factor Model

$$\xi = X\eta + F\varepsilon$$



- F diagonal
- η_i, ε_i in $N(0,1)$
- ξ may model
  - the times to default of n companies
  - the returns of n assets
- η are factors effecting all assets
- ε_i models movements only related to asset *i*.

Factor Model Applications

- Arises in modeling:
  - credit risk to rate collaterized debt obligations (CDOs)
  - capital asset for pricing (using multivariate time series),
  - forward rates
- New routine, G02AE.

Factor Model

- The problem yields a correlation matrix that can be written as

$$C = XX^{\mathrm{T}} + \mathrm{diag}\left(I - XX^{\mathrm{T}}\right)$$

- where *I* is the identity matrix and *X* had *n* rows and *k* columns.
- Our aim is to find *X*, the factor loading matrix, that gives a nearest correlation matrix to an (approximate) correlation matrix *G*.

Factor Model

- G02AE applies a spectral projected gradient method to the modified problem

$$\min \left\| G - XX^{\mathrm{T}} + \mathrm{diag}\left(XX^{\mathrm{T}} - I\right) \right\|_F$$

- such that

$$\left\| x_i^{\mathrm{T}} \right\|_2 \leq 1, \ \text{ for } \ i = 1, 2, \ldots, n,$$

- where x_i is the *i*th row of the factor loading matrix.

RECENT NEWS AND COMING SOON

New in the NAG Library (FL, CL, Toolbox, SMP)

- Complex Lambert W function
- Wavelet Transforms
  - One dimensional continuous transforms
  - Two dimensional discrete and multi-level transforms
- ODE's
  - BVP solution through Chebyshev psuedo-spectral method
- Matrix Operations
  - Matrix exponentials
  - Functions of real symmetric and Hermitian matricies
  - LAPACK 3.2 Cholesky solvers and factorizations
- Interpolation
  - Modified Shepard's method in 4D/5D
- Optimization
  - Minimization by quadratic approximation (BOBYQA)
  - Stochastic global optimization using PSO (most effective in the SMP library)
- RNG's
  - Generators of multivariate copulas
  - Skip-ahead for Mersenne Twister
  - L'Ecuyer MRG32K3a generator
- Statistics
  - Quantiles of streamed data, bivariate Student's t, and two probability density functions
  - Nearest correlation matrices
  - Hierarchical mixed effects regression
  - Quantile regression
  - Peirce Outlier detection
  - Anderson–Darling goodness-of-fit

New in... *continued*

- Optimization
  - Multi-start
  - Linear and non-linear SDP solvers (Semidefinate programming)
- Monte Carlo
  - Brownian Bridge constructor
  - Multi-level monte-carlo
- Special functions
  - 1F1, 2F1
- Vectorised functions
  - Misc statistical and special functions
- Long Names
- Improved support for R
  - R package (more than wrappers) for entire Optimisation chapter

Training and Consulting Services

- Product training
  - Libraries, Toolbox
- HPC and GPU Training
  - MPI, OpenMP
  - CUDA, OpenCL
- Flexible to fit needs, hands-on exercises

- Bespoke client work

NAG solutions for .NET

1. Call NAG C (or Fortran) DLL from C#
2. NAG Library for .NET **Launched Autumn 2010**
   - "a more natural solution"
   - DLL with C# wrappers, Integrated help
   - Includes the most popular NAG chapters (not the entire NAG Library)
   - Very popular with .NET developer community inc. in Finance Services
3. NAG Library for .NET (consultancy based)
   - as above pure C# functions

NAG routines for GPUs

- Monte Carlo components (available now)
  - L'Ecuyer mrg32k3a and Mersenne Twister (with skip-ahead) mt19937
  - Sobol sequence for Quasi-Monte Carlo (up to 50,000 dimensions)
  - Scrambled sequencing for Sobol (Hickernell)
  - Brownian Bridge
- *PDEs (coming soon)*
  - Finite Difference: Alternating Direction Implicit (ADI)
- Stochastic Volatility *(under investigation)*
  - Heston Model (again PDE based)
- ....

Summary

- Suite of routines coming for Matrix Functions
- Routines here now for Nearest Correlation Matrix problems. Improved functionality and performance coming soon.

- Numerical Libraries, Toolboxes and Services for you.
- You drive our development priorities.