## Week 6: Monday, Sep 28

# Logistics

1. Homework 1 is graded; the grades are in the CMS, and I have your work ready to pick up (assuming you turned in a hardcopy — otherwise, you should have received an email).

2. I updated the modified Cholesky code in problem 4; please use the newest version. Also, test harnesses for problems 2–4 are now posted on the assignments page.

A few notes from homework 1:

1. I forgot for a little bit that I'd made the modified code in problem 4 an optional/extra credit piece, so you can ignore any comments in which I asked why the code wasn't there. Each problem is out of four points, except problem 4, which is worth two points for the basic analysis and two points for the modified code.

2. Operators in MATLAB that start with a dot usually work componentwise. For example, `x .* y` refers to the vector with components $x_i y_i$; `x.^2` refers to a vector with components $x_i^2$; and so forth. It is generally more efficient to use these operators than to construct vectors and matrices with explicit loops.

3. MATLAB provides several log-scale plotting routines: `semilogy` and `semilogx` use a linear scale for one axis and a log scale for the other, and `loglog` uses a log scale for both axes. Log-scale plots are particularly useful for plotting errors, for example, since otherwise it tends to be difficult to see the details of the error after a certain point. So for the last problem on HW 1, for example, I plotted results using `semilogy(abs(s-pi))` (or equivalently with `semilogy(1:length(s), abs(s-pi))`.

4. Be aware that some of us have poor color vision, and may have a hard time distinguishing different curves based solely on color. Using different markers or line styles helps a lot.

5. For those of you using MATLAB to produce plots and then including them in LaTeX documents, I recommend writing out the MATLAB plots in PDF or (color) EPS form. These tend to be smaller than the corresponding PNG or JPG files, and they also tend to look better.

6. For MATLAB code listings, I recommend the `verbatim` package and the corresponding `verbatiminput` command. The `listings` package will produce nicely pretty-printed listings of MATLAB code, but I tend to prefer the teletype-style `verbatim` output.

# Symmetric matrices

A matrix $A$ is symmetric if $A = A^T$. For each symmetric matrix $A$, there is an associated quadratic form $x^T Ax$. You're likely familiar with quadratic forms from a multivariate calculus class, where they appear in the context of the second derivative test. One expands

$$F(x+u) = F(x) + F'(x)u + \frac{1}{2}u^T H(x)u + O(\|u\|^3),$$

and notes that at a stationary point where $F'(x) = 0$, the dominant term is the quadratic term. When $H$ is *positive definite* or *negative definite*, $x$ is a strong local minimum or maximum, respectively. When $H$ is indefinite, with both negative and positive eigenvalues, $x$ is a saddle point. When $H$ is semi-definite, one has to take more terms in the Taylor series to determine whether the point is a local extremum.

If $B$ is a nonsingular matrix, then we can write $x = By$ and $x^T Ax = y^T(B^T AB)y$. So an "uphill" direction for $A$ corresponds to an "uphill" direction for $B^T AB$; and similarly with downhill directions. More generally, $A$ and $B^T AB$ have the same *inertia*, where the inertia of a symmetric $A$ is the triple

$$(\#\text{ pos eigenvalues}, \#\text{ zero eigenvalues}, \#\text{ neg eigenvalues}).$$

Now suppose that $A = LU$, where $L$ is unit lower triangular and $U$ is upper triangular. If we let $D$ be the diagonal part of $U$, we can write $A = LDM^T$, where $L$ and $M$ are both unit lower triangular matrices. Noting that $A^T = (LDM^T)^T = MDL^T = M(LD)^T$ and that the $LU$ factorization

of a matrix is unique, we find $M = L$ and $LD = DM^T = U$. Note that $D$ has the same inertia as $A$.

The advantage of the $LDL^T$ factorization over the $LU$ factorization is that we need only compute and store one triangular factor, and so $LDL^T$ factorization costs about half the flops and storage of $LU$ factorization. We have the same stability issues for $LDL^T$ factorization that we have for ordinary $LU$ factorization, so in general we might compute

$$PAP^T = LDL^T,$$

where the details of various pivoting schemes are described in the book.

# Symmetric positive definite matrices (SPD matrices)

A symmetric matrix is positive definite if $x^T Ax > 0$ for all nonzero $x$. If $A$ is symmetric and positive definite, then $A = LDL^T$ where $D$ has all positive elements (because $A$ and $D$ have the same inertia). Thus, we can write $A = (LD^{1/2})(LD^{1/2})^T = \hat{L}\hat{L}^T$. The matrix $\hat{L}$ is a Cholesky factor of $A$.

The algorithm to compute the Cholesky factor of an SPD matrix is close to the Gaussian elimination algorithm. In the first step, we would write

$$\begin{bmatrix} a_{11} & a_{21}^T \\ a_{21} & A_{22} \end{bmatrix} = \begin{bmatrix} l_{11} & 0 \\ l_{21} & L_{22} \end{bmatrix} \begin{bmatrix} l_{11} & l_{21}^T \\ 0 & L_{22}^T \end{bmatrix},$$

or

$$\begin{aligned} a_{11} &= l_{11}^2 \\ a_{21} &= l_{21} l_{11} \\ A_{22} &= L_{22} L_{22}^T + l_{21} l_{21}^T. \end{aligned}$$

The first two equations allow us to compute the first column of $L$; the last equation tells us that the rest of $L$ is the Cholesky factor of a Schur complement, $L_{22} L_{22}^T = A_{22} - l_{21} l_{21}^T$. Continuing in this fashion, we have the algorithm

```
for j = 1:n
```

```
    % Compute the jth column of L
    A(j,j) = sqrt(A(j,j));
    A(j+1:end,j) = A(j+1:end,j)/A(j,j);

    % Update trailing submatrix
    A(j+1:end,j+1:end) = A(j+1:end,j+1:end) - ...
                         A(j+1:end,j)*A(j+1:end,j)';
end

% Lower triangle was overwritten by L
L = tril(A);
```

Like the nearly-identical Gaussian elimination algorithm, we can rewrite the Cholesky algorithm in block form for better cache use. Unlike Gaussian elimination, we do just fine using Cholesky *without* pivoting. To show this, we will need several related facts, which we shall discuss in more detail next time:

1. Any minor of an SPD matrix is SPD, and if $M$ is a minor of $A$, $\kappa_2(M) \leq \kappa_2(A)$.
2. The inverse of an SPD matrix is SPD.
3. Any Schur complement of an SPD matrix is SPD, and if $S$ is a Schur complement in $A$, $\kappa_2(S) \leq \kappa_2(A)$.