Notes

- Assignment 1 due tonight (email me by tomorrow morning)

The Power Method

- Start with some random vector $v$, $||v||_2=1$
- Iterate $v=(Av)/||Av||$
- The eigenvector with largest eigenvalue tends to dominate
- How fast?
  - Linear convergence, slowed down by close eigenvalues

Shift and Invert (Rayleigh Iteration)

- Say the eigenvalue we want is approximately $\lambda_k$
- The matrix $(A-\lambda_kI)^{-1}$ has the same eigenvectors as $A$
- But the eigenvalues are $\mu = \frac{1}{\lambda - \lambda_k}$
- Use this in the power method instead
- Even better, update guess at eigenvalue each iteration: $\lambda_{k+1} = v_{k+1}^T Av_{k+1}$
- Gives cubic convergence! (triples the number of significant digits each iteration when converging)

Maximality and Orthogonality

- Unit eigenvectors $v_1$ of the maximum magnitude eigenvalue satisfy $A v_1^2 = \max u=1 A u^2$
- Unit eigenvectors $v_k$ of the $k$th eigenvalue satisfy $||A v_k||_2 = \max_{\mu \neq \lambda_k} ||A u||_2$
- Can pick them off one by one, or….

Orthogonal iteration

- Solve for lots (or all) of eigenvectors simultaneously
- Start with initial guess $V$
- For $k=1, 2, \ldots$
  - $Z=AV$
  - $VR=Z$ (QR decomposition: orthogonalize $Z$)
- Easy, but slow (linear convergence, nearby eigenvalues slow things down a lot)

Rayleigh-Ritz

- Aside: find a subset of the eigenpairs
  - E.g. largest $k$, smallest $k$
- Orthogonal estimate $V (n \times k)$ of eigenvectors
- Simple Rayleigh estimate of eigenvalues:
  - $\text{diag}(V^T AV)$
- Rayleigh-Ritz approach:
  - Solve $k \times k$ eigenproblem $V^T AV$
  - Use those eigenvalues (Ritz values) and the associated orthogonal combinations of columns of $V$
- Note: another instance of "assume solution lies in span of a few basis vectors, solve reduced dimension problem"
Solving the Full Problem

- Orthogonal iteration works, but it's slow
- First speed-up: make A tridiagonal
  - Sequence of symmetric Householder reflections
  - Then Z=AV runs in O(n²) instead of O(n³)
- Other ingredients:
  - Shifting: if we shift A by an exact eigenvalue, A-λI, we get an exact eigenvector out of QR (the last column)
  - Division: once an off-diagonal is almost zero, problem separates into decoupled blocks

Nonlinear optimization

- Switch gears a little: we've already seen plenty of instances of minimizing, with linear least-squares
- What about nonlinear problems?

Classes of methods

- Only evaluate f:
  - Stochastic search, pattern search, cyclic coordinate descent (Gauss-Seidel), genetic algorithms, etc.
- Also evaluate ∂f/∂x (gradient vector)
  - Steepest descent and relatives
  - Quasi-Newton methods
- Also evaluate ∂²f/∂x² (Hessian matrix)
  - Newton's method and relatives

Steepest Descent

- The gradient is the direction of fastest change
  - Locally, f(x+dx) is smallest when dx is in the direction of negative gradient ∇f
- The algorithm:
  - Start with guess \( x^{(0)} \)
  - Until converged:
    - Find direction \( d^{(k)} = -\nabla f(x^{(k)}) \)
    - Choose step size \( \alpha^{(k)} \)
    - Next guess is \( x^{(k+1)} = x^{(k)} + \alpha^{(k)} d^{(k)} \)

Convergence?

- At global minimum, gradient is zero:
  - Can test if gradient is smaller than some threshold for convergence
  - Note: scaling problem: \( \min A^*f(B^*x)+C \)
- However, gradient is also zero at
  - Every local minimum
  - Every local maximum
  - Every saddle-point

Convexity

- A function is convex if
  \[
  f(\alpha x + (1-\alpha)y) \leq \alpha f(x) + (1-\alpha)f(y)
  \]
  \( \alpha \in [0,1] \)
- Eliminates possibility of multiple strict local mins
- Strictly convex: at most one local min
- Very good property for a problem to have!
Selecting a step size

- Scaling problem again: physical dimensions of x and gradient may not match
- Choosing a step too large:
  - May end up further from minimum
- Choosing a step too small:
  - Slow, maybe too slow to actually converge

- **Line search**: keep picking different step sizes until satisfied