Asymptotic Behavior $f(t)$ is amyptotic to t means the following:

$$f(t) \sim t^2 \text{ as } t \to 0 \text{ means } t^{-2}f(t) \to 0 \text{ as } t \to 0$$

Equivalently, we write $f(t) = t^2 + o(t^2)$. Also, it could be the case:

$$f(t) = O(t^2) \text{ means } t^{-2}f(t) \text{ is bounded as } t \to 0$$

1 Floating Point Arithmetic

A **Floating Point Number System** is a finite subset of the reals defined by $\mathbb{F}(b, K, m, M)$ where b is the base of the system, K is the number of digits, m is the smallest exponent representable and M is the largest exponent representable.
If $y \in \mathbb{F}(b, K, m, M)$, then

$$y = \pm\left(0.d_1d_2d_3....d_K\right)_b \times b^E, m \leq M, d_1 \neq 0 \iff y = 0$$

1.1 Round-off Error

The error in representing $z \in \mathbb{R}$ by its nearest element in $\mathbb{F}(b, K, m, M)$. If $z \in \mathbb{R}$, then

$$fl(z) = \begin{cases} \pm(0.d_1d_2...d_K)_b \times b^E & d_{k+1} < \frac{b}{2} \\ \pm\left[(0.d_1d_2...d_K)_b + b^{-K}\right] \times b^E & d_{k+1} \geq \frac{b}{2} \end{cases}$$

Example IEEE Double precision (Used in MATLAB) is $b = 2$ and $K = 52$. In base 10, this is approximately $K \approx 16$, $m \approx -308$, and $M \approx 308$.

1.2 Relative error in rounding

Let $y \in \mathbb{R}, y \neq 0.$ $fl(y) \in \mathbb{F}(b, K, m, M)$. Assume $d_{k+1} \leq \frac{b}{2}$

$$|fl(y)| = (0.d_1d_2...d_K)_b \times b^E$$

The **Relative error** is

$$\text{Rel error: } \frac{|y - fl(y)|}{|y|}$$

Since $|y| = (0.d_1d_2...d_kd_{k+1}...)_b \times b^E \geq (0.1)_b \times b^E = b^{E-1}$
and $|y - fl(y)| = (0.d_{k+1 d_{k+2}....b} \times b^{E-k} \leq \frac{1}{2}b^{E-k}$ thus

$$\text{Rel error: } \frac{|y - fl(y)|}{|y|} \leq \frac{1}{2}b^{1-K} = \epsilon_{machine}$$

This **Machine epsilon** is the smallest representable number

Example In IEEE DP, $b = 2$, and $K = 52$, so $_{\text{machine}} = 2^{-52} \approx 2.2204 \times 10^{-16}$

1.3 Error in Computation

1.3.1 Finite Difference Operators

Recall

$$f''(x) - \frac{1}{h^2}\left(f(x-h) - 2f(x) + f(x+h)\right) = -\frac{1}{12}h^2 f^{(4)}(\xi)$$

Assume the round off error $e(x-h)$ is in the evaluation of function values $f(x-h) = \tilde{f}(x-h) + e(x-h)$.

$$f''(x) - \frac{1}{h^2}\left(\tilde{f}(x-h) - 2\tilde{f}(x) + \tilde{f}(x+h)\right) = E_h = -\frac{1}{12}h^2 f^{(4)}(\xi) + \frac{1}{h^2}\left(e(x-h) - 2e(x) + e(x+h)\right)$$

$$|E_h| \leq \frac{1}{12}h^2 \left|f^{(4)}(\xi)\right| + \frac{1}{h^2}\left(|e(x-h)| + |2e(x)| + |e(x+h)|\right)$$

Assume $\left|f^{(4)}(\xi)\right| \leq M$ for $\xi \in [x-h, x+h]$ and assume $|e(x)| \leq \epsilon$ for $x \in [x-h, x+h]$.

$$|E_h| \leq \frac{1}{12}h^2 M + \frac{1}{h^2}4\epsilon$$

The first term shrinks but the second term blows up as $h \to 0$. One hopes to find the minimum at

$$h_{optimal} = \left(\frac{48\epsilon}{M}\right)^{\frac{1}{4}}$$

We could take ϵ to be $\epsilon_{machine}$

2 Polynomial Approximations

2.1 Taylor Expansion Theorem

The Taylor series expansion for a function f centered at α evaluated at z is

$$f(z) = \sum_{k=0}^{\infty} a_k (z-\alpha)^k \qquad \text{where } a_k = \frac{f^{(k)}(\alpha)}{k!}$$

A Taylor polynomial is any finite truncation of this series:

$$f(z) = \sum_{k=0}^{N} a_k (z-\alpha)^k \qquad \text{where } a_k = \frac{f^{(k)}(\alpha)}{k!}$$

The Taylor series is the limit of the Taylor Polynomials, given that the limit exists.

Analytic Functions A function that is equal to its Taylor Series in an open interval (or open disc in the complex plane), is known as an **Analytic Function**

Maclaurin Series If the Taylor series or Polynomial is centered at the origin ($\alpha = 0$), then it is also a MacLaurin series.

2.1.1 Important Taylor Series

The Maclaurin series for $(x-1)^{-1}$ is

$$(x-1)^{-1} = 1 + x + x^2 + x^3 + ... = \sum_{k=0}^{\infty} x^k$$

3 Numerical Linear Algebra

4 Solving $Ax = b$

4.1 Tridiagonal Solver

For a tridiagonal system of equations

$$A\vec{\mathbf{u}} = \vec{\mathbf{f}}, \qquad A \text{ tridiagonal}$$

take $b_1 = c_n = 0$ and

$$A = LU = \begin{pmatrix} a_1 & c_1 & & \\ b_2 & a_2 & c_2 & \\ & \ddots & \ddots & \ddots \\ & & b_n & a_n \end{pmatrix} = \begin{pmatrix} 1 & & & \\ \beta_2 & 1 & & \\ & \ddots & \ddots & \\ & & \beta_n & 1 \end{pmatrix} \begin{pmatrix} \alpha_1 & c_1 & & \\ & \alpha_2 & c_2 & \\ & & \ddots & \ddots \\ & & & \alpha_n \end{pmatrix}$$

So to solve $LU\vec{\mathbf{u}} = \vec{\mathbf{f}}$

1. Solve $L\vec{\mathbf{v}} = \vec{\mathbf{f}}$ using Forward Substitution
2. Solve $U\vec{\mathbf{u}} = \vec{\mathbf{v}}$ using Backward Substitution

4.1.1 Pseudocode

```
INPUT: $\vec{\mathbf{a}}, \vec{\mathbf{b}}, \vec{\mathbf{c}}, \vec{\mathbf{f}}$, all length $n$ LU decomposition:
$\alpha_1 = a_1$
for $k = 2$ to $n$
    $\beta_k = b_k \setminus \alpha_{k-1}$
    $\alpha_k = a_k - \beta_k c_{k-1}$
end
Forward Substitution:
$v_1 = f_1$
for $k = 2$ to $n$
    $v_k = f_k - \beta_k v_{k-1}$
end
Backward Substitution:
$u_n = v_n \setminus \alpha_n$
for $k = 2$ to $n$
    $j = (n+1) - k$
    $u_j = (v_j - c_j u_{j+1}) \setminus \alpha_j$
end
Operation count: $O(n)$
```

4.2 Spectral Decomposition Method

If $A \in \mathbb{C}^{m \times m}$ is Hermitian, we can do the following

1. Compute the spectral decomposition (not trivial when m is large)

$$A = UDU^*$$

2. Reform equation

$$A\vec{\mathbf{x}} = UDU^*\vec{\mathbf{x}} = \vec{\mathbf{b}}$$

So we see

$$\vec{\mathbf{x}} = \alpha_1\vec{\mathbf{u}}_1 + \alpha_2\vec{\mathbf{u}}_2 + ... + \alpha_m\vec{\mathbf{u}}_m \implies U^*\vec{\mathbf{x}} = \vec{\alpha}$$

and

$$\vec{\mathbf{b}} = \beta_1\vec{\mathbf{u}}_1 + \beta_2\vec{\mathbf{u}}_2 + ... + \beta_m\vec{\mathbf{u}}_m \implies \beta_k = \vec{\mathbf{u}}_k^*\vec{\mathbf{b}}$$

so

$$U^*\vec{\mathbf{b}} = \vec{\beta} \text{ and } D\vec{\alpha} = \vec{\beta} \implies \vec{\alpha} = D^{-1}\vec{\beta}$$

but since $D_{ii}^{-1} = \frac{1}{\lambda_i}$,

$$\alpha_k = \frac{\vec{\mathbf{u}}_k^*\vec{\mathbf{b}}}{\lambda_k} \implies \vec{\mathbf{x}} = \sum_{k=1}^{m}\left(\frac{\vec{\mathbf{u}}_k^*\vec{\mathbf{b}}}{\lambda_k}\right)\vec{\mathbf{u}}_k$$

5 Finite Differences

Finite Differences seeks to approximate an ODE or PDE over a **mesh** or **grid**. The steps involved are:

1. Discretize the PDE using a difference scheme.
2. Solve the discretized PDE by iterating and/or time stepping.

5.1 Meshes

5.1.1 Uniform Meshes

Given a closed domain $\Omega = \bar{R} \times [0, t_F]$, we divide it into a $(J+1) \times (N+1)$ grid of parallel lines. Assume $\bar{R} = [0, 1]$. Given the mesh sizes $\Delta x = \frac{1}{J}, \Delta t = \frac{1}{N}$, a **mesh point** is

$$(x_j, t_n) = (j\Delta x, n\Delta t) \qquad j = 0, ..., J \qquad n = 0, ..., N$$

and $x_0 = 0$, $x_n = 1$
An alternative convention uses a $(J+2) \times (N+2)$ grid with the mesh sizes $\Delta x = \frac{1}{J+1}, \Delta t = \frac{1}{N+1}$. $x_0 = 0$, $x_{n+1} = 1$ are the boundary points. and

$$(x_j, t_n) = (j\Delta x, n\Delta t) \qquad j = 0, ..., J+1 \qquad n = 0, ..., N+1$$

We seek approximations to the solution at these mesh points, denoted by

$$U_j^n \approx u(x_j, t_n)$$

Where initial values are exact from the initial value function $u^0(x, t) = u(x, 0)$

$$U_j^0 = u^0(x_j) \qquad j = 1, ..., J-1$$

and boundary values are exact from the boundary value functions $f(t) = u(0, t)$ and $g(t) = u(1, t)$

$$U_0^n = f(t_n) \qquad U_J^n = g(t_n) \qquad n = 1, 2, ...,$$

5.2 Difference Coefficients

$$D_+, D_-$$

5.3 Explicit Scheme

A scheme is **explicit** if the solution at the next iteration (time level t_{n+1}) can be written as a single equation involving only previous time steps. This is, if it can be written in the form

$$U_j^{n+1} = \sum_i \sum_{k \le n} a_{i,k} U_i^k + b_{i,k} f_i^k$$

Example For the Heat Equation $u_t = u_{xx}$, using a forward difference in time and a centered difference in space, we get

$$U_j^{n+1} = U_j^n + \mu(U_{j+1}^n - 2U_j^n + U_{j-1}^n) \qquad \mu \coloneqq \frac{\Delta t}{(\Delta x)^2}$$

Pseudocode :

```
At $n = 0$, $U_j^0 = u^0(x_j, 0)$
for $n = 1 : N$
    $U_0^n = 0, U_J^n = 0$
    for $j = 1 : (J-1)$
        $U_j^{n+1} = U_j^n + \mu(U_{j+1}^n - 2U_j^n + U_{j-1}^n)$
    end
end
```

The **stability** of the problem depends on μ.

5.4 Truncation Error

Example For our model problem (the Heat Equation), the trucation error is

$$T(x,t) \coloneqq \frac{D_{+t}u(x,t)}{\Delta t} - \frac{D_x^2 u(x,t)}{(\Delta x)^2}$$

So we see that since $u_t - u_{xx} = 0$,

$$T(x,t) = (u_t - u_{xx}) + \left(\frac{1}{2}u_{tt}\Delta t - \frac{1}{12}u_{xxxx}(\Delta x)^2\right) + \ldots = \left(\frac{1}{2}u_{tt}\Delta t - \frac{1}{12}u_{xxxx}(\Delta x)^2\right) + \ldots$$

If we truncate this Infinite Taylor series using $\eta \in (t, t + t\Delta t)$ and $\xi \in (x - \Delta x, x + \Delta x)$ and assume the boundry and initial data are consistent at the corners and are both sufficientl smooth, we can then estimate $|u_{tt}(x,\eta)| \le M_{tt}$ and $|u_{xxxx}(\xi, n)| \le M_{xxxx}$, so it follows that

$$|T(x,t)| \le \frac{1}{2}\Delta t\left(M_{tt} - \frac{1}{6\mu}M_{xxxx}\right)$$

We can assume these bounds will hold uniformly over the domain. We see that

$$|T(x,t)| \to 0 \text{ as } \Delta t, \Delta x \to 0 \;\forall\; (x,t) \in \Omega$$

and this result is independent of any relaton between the two mesh sizes. Thus this scheme is **unconditionally consistent** with the differential equation.

Since $|T(x,t)|$ will behave asymptotically like $O(\Delta t)$ as $\Delta t \to 0$, this scheme is said to have **first order accuracy**

Since $u_t = u_{xx}$, $u_{tt} = u_{xxxx}$ and so for $\mu = \frac{1}{6}$,

$$T(x,t) = \frac{1}{2}\Delta t\left(u_{tt} - \frac{1}{6\mu}u_{xxxx}\right) + O\left((\Delta t)^2\right) = O\left((\Delta t)^2\right)$$

and so the scheme is **second order accurate** for $\mu = \frac{1}{6}$

We can define notation: $T_j^n = T(x_j, t_n)$

5.5 Consistency

Does the difference scheme approximate the PDE as $\Delta x, \Delta t \to 0$?
A scheme is consistent if

$$T(x,t) \to 0 \text{ as } \Delta x, \Delta t \to 0$$

- A scheme is **unconditionally consistent** if the scheme is consistent for any relationship between Δx and Δt
- A scheme is **conditionally consistent** if the scheme is consistent only for certain relationships between Δx and Δt

5.6 Accuracy

Take μ finite, so $(\Delta t)^{\frac{a}{b}} = \mu(\Delta x)^{\frac{c}{d}}$, so $(\Delta t)^{\alpha} = (\Delta t)^{ad} = \mu^{bd}(\Delta x)^{cb}$ and we have

$$|T(x,t)| = O(\Delta t^{\alpha})$$

- If $\alpha = 1$, the scheme is **first order accurate**.
- If $\alpha = 2$, the scheme is **second order accurate**.
- etc...
- The scheme is α**-order accurate**.

5.7 Convergence

A scheme is **convergent** if as $\Delta t, \Delta x \to 0$ for any fixed point (x^*, t^*) in the domain,

$$x_j \to x^*, t_n \to t^* \implies U_j^n \to u(x^*, t^*)$$

It suffices to show this for mesh points for sufficiently refined meshes, as convergence at all other points will follow from continuity of $u(x,t)$. We suppose that we can find a bound for the error $\bar{T}$:

$$\left|T_j^n\right| \leq \bar{T} < \infty$$

We denote the **error**

$$e_j^n := U_j^n - u(x_j, t_n)$$

Taking the difference between the scheme and $u(x_j, t^{n+1})$ in terms the truncation error and the exact solution at previous time steps yields the error at e_j^{n+1}. If the RHS of our difference scheme is represented by D, then

$$e_j^{n+1} = DU_j^n - (Du(x_j, t_n) + T(x_j, t_n)\Delta t) = De_j^n - T_j^n \Delta t$$

Choose μ such that the coefficients of the RHS are positive so that you may estimate $E^n := \max\left\{\left|e_j^n\right|, j = 0, ..., J\right\}$ and so

$$E^{n+1} \leq E^n + \bar{T}\Delta t \text{ s.t. } E^0 = 0$$

and thus

$$E^n \leq n\bar{T}\Delta t \qquad n = 0, 1, 2, ...$$

and considering the domain

$$E^n \leq \bar{T}t_F \qquad n = 0, 1, 2, ..., N$$

and since $\bar{T} \to 0$ as $\Delta t, dx \to 0$, $E^n \to 0$

Example If we replace U_j^n with $u(x_j, t_n)$ in the definition of T_j^n we obtain

$$e_j^{n+1} = e_j^n + \mu D_x^2 e_j^n - T_j^n \Delta t$$

which is

$$e_j^{n+1} = (1-2\mu)e_j^n + \mu e_{j+1}^n + \mu e_{j-1}^n - T_j^n \Delta t$$

For $\mu \leq \frac{1}{2}$, define $E^n := \max\left\{\left|e_j^n\right|, j = 0, ..., J\right\}$ and so

$$\left|e_j^{n+1}\right| \leq E^n + \bar{T}\Delta t \implies E^{n+1} \leq E^n + \bar{T}\Delta t$$

Since $E^0 = 0$ (the initial values are exact), we have

$$E^n \leq n\bar{T}\Delta t = \frac{1}{2}\Delta t \left(M_{tt} - \frac{1}{6\mu}M_{xxxx}\right) t_F \to 0 \text{ as } t \to 0$$

5.7.1 Refinement Path

A **refinement path** is a sequence of pairs of mesh sizes each which tends to zero

$$\text{refinement path} := \{((\Delta x)_i, (\Delta t)_i), i = 0, 1, 2, ...; (\Delta x)_i, (\Delta t)_i \to 0\}$$

We can specify particular paths by requiring certain relationships between the mesh sizes.

Examples $(\Delta t)_i \sim (\Delta x)_i$ or $(\Delta t)_i \sim (\Delta x)_i^2$

Theorem For the heat equation, $\mu_i = \frac{(\Delta t)_i}{(\Delta x)_i^2}$ and if $\mu_i \leq \frac{1}{2}\ \forall\ i$ and if for all sufficiently large values of i and the positive numbers n_i, j_i are such that

$$n_i(\Delta t)_i \to t > 0, j_i(\Delta x)_i \to x \in [0, 1]$$

and if $|u_{xxxx}| \leq M_{xxxx}$ uniformly on Ω, then the approximations $U_{j_i}^{n_i}$ generated by the explicit scheme for $i = 0, 1, ...$ converge to the solution $u(x, t)$ of the differential equation uniformly in the region.
This means that arbitrarily good accuracy can be attained by use of a sufficiently fine mesh.

5.8 Error: Fourier Analysis

Let

$$U_j^n = (\lambda)^n e^{ik(j\Delta x)}$$

where $\lambda(k)$ is known as the **amplification factor** of the **Fourier Node** U_j^n. Place this into the difference equation of your scheme and solve for λ. We then have another numerical approximation

$$U_j^n = \sum_{-\infty}^{\infty} A_m e^{-im\pi(j\Delta x)} (\lambda(k))^n$$

which can be compared to the Fourier expansion approximating the exact solution.

Example For $U_j^n = U_j^n + \mu(U_{j+1}^n - 2U_j^n + U_{j-1}^n)$, we see

$$\lambda(k) = 1 + \mu(e^{ik\Delta x} - 2 + e^{-ik\Delta x}) = 1 - 4\mu \sin^2\left(\frac{1}{2}k\Delta x\right)$$

So now

$$e^{-k^2\Delta t} - \lambda(k) = \left(1 - k^2\Delta t + \frac{1}{2}k^4\Delta t(\Delta t)^2 - ...\right) - \left(1 - k^2\Delta t + \frac{1}{12}k^4\Delta t(\Delta x)^2 - ...\right) = \left(\frac{(\Delta t)^2}{2} - \frac{\Delta t(\Delta x)^2}{12}\right)k^4 - ...$$

Thus we have first order accuracy in general but second order accuracy if $(\Delta x)^2 = 6\Delta t$.

5.9 Stability

A scheme is **stable** if there exists a constant K such that

$$|(\lambda)|^n \leq K, \qquad n\Delta t \leq t_F, \; \forall \, k$$

That is, if the difference in the solutions of the DE and the numerical DE is bounded uniformly in the domain for any amount of time less than t_F. Thus

$$|\lambda(k)| \leq 1 + K'\Delta t$$

This is necessary and sufficient.

5.10 Implicit Scheme

If the scheme cannot be written in a form that has U_j^{n+1} explicitly computed given values U_j^n, $j = 0, 1, ..., J$, it is implicit. Implicit schemes involve more work but often have higher accuracy and/or stability, and thus much larger time steps allow us to reach the solution much more quickly.

Example

$$\frac{U_j^{n+1} - U_j^n}{\Delta t} = \frac{U_{j+1}^{n+1} - 2U_j^{n+1} + U_{j-1}^{n+1}}{(\Delta x)^2}$$

which can be written as

$$\Delta_{-t} U_j^{n+1} = \mu \delta_x^2 U_j^{n+1} \qquad \mu = \frac{\Delta t}{(\Delta x)^2}$$

This involves solving a system of linear equations. However, Fourier analysis for the stability shows

$$\lambda = \frac{1}{1 + 4\sin^2\left(\frac{1}{2}k\Delta t\right)}$$

Since $\lambda < 1$ for any positive μ, this scheme is **unconditionally stable**

5.11 Other Conditions

If an equation obeys extra conditions such as a Maximum Principle, uniqueness condition, or a physical constraint, the numerical scheme must also obey such conditions else it may not converge.

6 Methods

6.1 Weighted Average θ method

Given two schemes, you can weight one θ and the other with $(1-\theta)$ and add them together. Then stability, covergence, and accuracy may depend on θ, and it can be chosen to

Example For the explicit and implicit first order accurate schemes for the heat equation are averaged, we have

$$U_j^{n+1} - U^n = \mu\left(\theta\delta_x^2 U_j^{n+1} + (1-\theta)\delta_x^2 U_j^n\right)$$

$\theta = 0$ yields the explicit scheme and $\theta = 1$ yields the implicit scheme.

7 General Boundary Conditions

Boundary conditions like

$$u_x = \alpha(t)u + g(t) \qquad x = 0$$

Can be handled like

$$\frac{U_1^n - U^n)0}{\Delta x} = \alpha^n U_0^n + g^n \implies U_0^n = \beta^n U_1^n - \beta^n g^n \Delta t \qquad \beta^n = \frac{1}{1 + \alpha^n \Delta x}$$

Dirichlet conditions are trivial

$$u(0,t) = 0 \implies U_0^n = 0$$

$$D_x^2 y_i = D_+ D_- y_i = \frac{1}{h^2}(y_{i+1} - 2y_i + y_{i-1})$$

$$y_{i\pm1} = y_i \pm hy_i' + \frac{1}{2}h^2 y_i'' \pm \frac{1}{6}h^3 y_i''' + \frac{1}{24}h^4 y_i''''...$$

$$D_x^2 y_i = \frac{1}{h^2}\left(y_i + hy_i' + h^2 y'' + \frac{1}{6}h^3 y_i''' + \frac{1}{24}h^4 y_i'''' - 2y_i + y_i - hy_i' + h^2 y'' - \frac{1}{6}h^3 y_i''' + \frac{1}{24}h^4 y_i'''' + ...\right)$$

which simplifies to

$$D_x^2 y_i = y_i'' + O(h^2)$$

Let $u_i \approx y_i = y(x_i)$

$$u_{xx} + q(x)u = f(x)$$

with $u(0) = \alpha$ and $u(1) = \beta$ becomes

$$\frac{-1}{h^2}(u_{i+1} - 2u_i + u_{i-1}) + q_i u_i = f_i$$

or

$$\begin{cases} -u_2 + (2 + h^2 q_1)u_1 = h^2 f_1 + \alpha & i = 1 \\ -u_{i+1} + (2 + h^2 q_i)u_i - u_{i-1} = h^2 f_i & 2 \le i \le n \\ (2 + h^2 q_n)u_n - u_{n-1} = h^2 f_n + \beta & i = n \end{cases}$$

where $u_0 = \alpha$ and $u_{n+1} = \beta$ we must solve

$$A_n \vec{\mathbf{u}}_n = \begin{pmatrix} 2 + h^2 q_1 & -1 & & \\ -1 & 2 + h^2 q_2 & -1 & \\ & \ddots & \ddots & \ddots \\ & & -1 & 2 + h^2 q_n \end{pmatrix} \begin{pmatrix} u_1 \\ \vdots \\ \vdots \\ u_n \end{pmatrix} = \begin{pmatrix} h^2 f_1 + \alpha \\ h^2 f_2 \\ \vdots \\ h^2 f_n + \beta \end{pmatrix} = \vec{\mathbf{f}}_n$$

Note: A_n is tridiagonal and symmetric. We can solve this by using $A_n = LU$

$$L\vec{\mathbf{v}}_n = \vec{\mathbf{f}}_n \qquad U\vec{\mathbf{u}}_n = \vec{\mathbf{v}}_n$$

8 New Notes

9 Finite Difference Coefficients

9.1 Centered Differences

For the difference scheme for the α derivative of f,

$$f^{(\alpha)}(x_i) \approx D_h^\alpha f = \frac{1}{d}\frac{a_{i-4}f(x_{i-4}) + ... + a_i f(x_i) + ... + a_{i+4}f(x_{i+4})}{h^\alpha}$$

where d is the denominator to make the coefficients a_i integers. If we want the scheme to have accuracy β,

$$f^{(\alpha)}(x_i) = D_h^\alpha f + O(h^\beta)$$

Then the difference coefficients are given by

α	β	d	a_{i-4}	a_{i-3}	a_{i-2}	a_{i-1}	a_i	a_{i+1}	a_{i+2}	a_{i+3}	a_{i+4}
1	2	2				-1	0	1			
	4	12			1	-8	0	8	-1		
	6	60		-1	9	-45	0	45	-9	1	
	8	840	3	-32	168	-672	0	672	-168	32	-3
2	2	1				1	-2	1			
	4	12			-1	16	-30	16	-1		
	6	180		2	-27	270	-490	270	-27	2	
	8	5040	-9	128	-1008	8064	-14350	8064	-1008	128	-9
3	2	2			-1	2	0	-2	1		
	4	8		1	-8	13	0	-13	8	-1	
	6	240	-7	72	-338	488	0	-488	338	-72	7
4	2	1			1	-4	6	-4	1		
	4	6		-1	12	-39	56	-39	12	-1	
	6	240	7	-96	676	-1952	2730	-1952	676	-96	7

9.2 Forward/Backwards Differences

For the difference scheme for the α derivative of f,

$$f^{(\alpha)}(x_i) \approx D_\pm^\alpha f = \frac{1}{d}\frac{a_i f(x_i) + ... + a_{i\pm4}f(x_{i\pm4}) + ... + a_{i\pm8}f(x_{i\pm8})}{h^\alpha}$$

where d is the denominator to make the coefficients a_i integers. If we want the scheme to have accuracy β,

$$f^{(\alpha)}(x_i) = D_\pm^\alpha f + O(h^\beta)$$

Then the difference coefficients are given by

α	β	d	a_i	a_{i+1}	a_{i+2}	a_{i+3}	a_{i+4}	a_{i+5}	a_{i+6}	a_{i+7}	a_{i+8}
1	1	1	∓ 1	± 1							
	2	2	∓ 3	± 4	∓ 3						
	3	6	∓ 11	± 18	∓ 9	± 2					
	4	12	∓ 25	± 48	∓ 36	± 16	∓ 3				
	5	60	∓ 137	± 300	∓ 300	± 200	∓ 75	± 12			
	6	60	∓ 147	± 360	∓ 450	± 400	∓ 225	± 72	∓ 10		
2	1	1	1	-2	1						
	2	1	2	-5	4	-1					
	3	12	35	-104	114	-56	11				
	4	12	45	-154	214	-156	61	-10			
	5	180	812	-3132	5265	-5080	2970	-972	137		
	6	180	938	-4014	7911	-9490	7389	-3616	1019	-126	
3	1	1	∓ 1	± 3	∓ 3	1					
	2	2	∓ 5	± 18	∓ 24	± 14	∓ 3				
	3	4	∓ 17	± 71	∓ 118	± 98	∓ 41	± 7			
	4	8	∓ 49	± 232	∓ 461	± 496	∓ 307	± 104	∓ 15		
	5	120	∓ 967	± 5104	∓ 11787	± 15560	∓ 12725	± 6432	∓ 1849	± 232	
	6	240	∓ 2403	± 13960	∓ 36706	± 57384	∓ 58280	± 39128	∓ 16830	± 4216	∓ 469
4	1	1	1	-4	6	-4	1				
	2	1	3	-14	26	-24	11	-2			
	3	6	35	-186	411	-484	321	-114	17		
	4	6	56	-333	852	-1219	1056	-555	164	-21	
	5	240	3207	-21056	61156	-102912	109930	-76352	33636	-8576	967

Part I
New Notes

10 Two Dimensional Problems

Given a problem

$$u_t = b(u_{xx} + u_{yy} \qquad \Omega = [0, X] \times [0, Y]$$

With initial conditions on Ω for $t = 0$ and boundary conditions on $\partial\Omega$

So for a uniform grid mesh

$$U_{i,j}^N \approx u(x_i, y_j, t_n) = u(i\Delta x, j\Delta y, n\Delta t), i = 0, 1, ...I, j = 0, 1, ...J, n = 0, 1, ...N$$

Explicit Scheme

$$\frac{U_{i,j}^{n+1} - U_{i,j}^n}{\Delta t} = b\left(\frac{\delta_x^2 U_{i,j}^n}{\Delta x^2} + \frac{\delta_y^2 U_{i,j}^n}{\Delta y^2}\right)$$

Consistency

$$T_{ij}^n = \left(\frac{1}{2}\Delta t u_{tt} - \frac{1}{12} b\left(\Delta x^2 u_{xxxx} + \Delta y^2 u_{yyyy}\right)\right)_{ij}^n + ...$$

$$T_{ij}^n \approx O(\Delta t + \Delta x^2 + \Delta y^2)$$

Stability

$$U_{ij}^n \approx (\lambda)^n e^{i(k_x i\Delta x + k_y i\Delta y)}$$

????

Convergence The error estimate

$$e_{ij}^n = U_{ij}^n - u_{ij}^n$$

$$U_{ij}^n = U_{ij}^n + b\left(\mu_x \delta_x^2 U_{ij}^n + \mu_y \delta_y^2 U_{ij}^n\right)$$

$$e_y^{n+1} = e_y^n + b\left(\mu_x \delta_x^2 e_{ij}^n + \mu_y \delta_y^2 e_{ij}^n\right) - \Delta t T_{ij}^n$$

$$\mu_x = b\frac{\Delta t}{\Delta x^2}, \mu_y = b\frac{\Delta t}{\Delta y^2}$$

10.0.1 θ Scheme

Accuracy is $O(\Delta t + \Delta x^2 + \Delta y^2)$, but if $\theta = \frac{1}{2}$ we have the Crank-Nicholson Scheme with accuracy $O(\Delta t^2 + \Delta x^2 + \Delta y^2)$.

This requires to solve a system

$$A\vec{\mathbf{U}}^{n+1} = \vec{\mathbf{b}}^n$$

Where $U^n ij$ is reshaped into a vector.

Modified Gram-Schmidt INPUT: $A \in \mathbb{C}^{m\times n}$ where $m \geq n$ and $\operatorname{rank}(A) = n$ OUTPUT: $\hat{Q} \in \mathbb{C}^{m\times n}$ orthogonal, $\hat{R} \in \mathbb{C}^{n\times n}$ upper triangular, where $A = \hat{Q}\hat{R}$

for i=1 to n r$_{ii} = \|\vec{\mathbf{a}}_i\|\ _2\vec{\mathbf{q}} = \frac{\vec{\mathbf{a}}_i}{r_{ii}} for j = i{+}1 to n r_{ij} = \vec{\mathbf{q}}_i^* \vec{\mathbf{a}}_j \vec{\mathbf{a}}_j = \vec{\mathbf{a}}_j - r_{ij}\vec{\mathbf{q}}_i end end Operation count flops \tilde{\sum}_{i=1}^n \sum_{j+1}^n (m + (m-1) + m + m = \sum_{i=1}^n \sum_{j+1}^n (4m-1) = (4m-1)\sum_{i=1}^n (n-i) = (4m-1)(n^2 - \frac{1}{2}n^2)\tilde{2} mn^2 (same as classical GS)$

- Modified GS performs better than Classical GS when there is roundoff error involved.
- Both run into issues with ill conditioned matrices.

The modified GS creates a sequence of matrices $\{R_n\}$

$$AR_1R_2...R_N = [\vec{\mathbf{q}}_1 \dots \vec{\mathbf{q}}_n] = \hat{Q}$$

$$\hat{R} = (R_1R_2...R_N)^{-1} = R_N^{-1}...R_2^{-1}R_1^{-1}$$

Triangular Orthogonalization: Post-multiplying A by a sequence of upper triangular matrices to produce an orthogonal matrix.

Householder Method: QR factorization Based on Orthogonal triangularization

$$Q_N...Q_2Q_1A = Q^{-1}A = R(fullQRfactorization)$$

Application of Least Squares: Approximating Data. Find a function $f(x)$, typically a polynomial with unknown coefficients, that approximates ome data set $(x_i, y_i)\ i = 1, 2, ..., N$ in the sense that $\|f(\vec{\mathbf{x}}) - \vec{\mathbf{y}}\|_2$ is minimized (we assume there is error in the data).
Assume $f(x) = a_0 + a_1x + ... + a_{n-1}x^{n-1}$ and $m \geq n$. Let $\vec{\mathbf{x}}_i^k = x_i^k$, $\vec{\mathbf{a}}_i = a_i$, and $\vec{\mathbf{b}}_i = y_i$ so

$$A \in \mathbb{R}^{n\times m}, A := \left(\vec{\mathbf{1}}, \vec{\mathbf{x}}, \vec{\mathbf{x}}^2, ..., \vec{\mathbf{x}}^{n-1}\right), \qquad A\vec{\mathbf{a}} = \vec{\mathbf{b}}$$

Problem: Find $\vec{\mathbf{a}}$ such that $\|\vec{\mathbf{r}}\|_2 = \|\vec{\mathbf{y}} - A\vec{\mathbf{a}}\|_2$ is minimized.

10.1 Solution using Normal Equations

A^*A is invertible as long as there are at least n distinct x_is...that is $rank(A) = \min(n,$quantity of distinct x_i's). So we can simply use

$$\vec{\mathbf{a}} = (A^*A)^{-1}A^*\vec{\mathbf{y}}$$

10.1.1 Implementation: Solving Normal Equations

Solve Normal Equations via Cholesky Factorization $A^*A = R^*R$ since A^*A is Hermitian and if $rank(A^*A) = n$ it is positive definite.

1. Form A^*A, and $A^*\vec{\mathbf{b}}$ ($\sim mn^2$ flops)
2. Form Cholesky Facotrization A^*A=R^*R ($\sim \frac{1}{3}n^3$ flops)
3. Solve $R^*\vec{\mathbf{y}} = A^*\vec{\mathbf{b}}$ via Forward substitution ($\sim n^2$ flops)
4. Solve $R\vec{\mathbf{a}} = \vec{\mathbf{y}}$ via Backward substitution ($\sim n^2$ flops)

Total operation count: $\sim mn^2 + \frac{1}{3}n^3$ flops

10.1.2 Implementation: QR Decomposition

1. Form reduced QR decomposition $A = \hat{Q}\hat{R}$ via Householder Triangularization ($\sim 2mn^2 - \frac{2}{3}n^3$ flops)
2. Form $\vec{\mathbf{y}} = \hat{Q}^*\vec{\mathbf{b}}$ ($\sim 2mn$)
3. Solve $\hat{R}\vec{\mathbf{a}} = \vec{\mathbf{y}}$ via Backward substitution ($\sim n^2$ flops)

Total operation count: $\sim 2mn^2 - \frac{2}{3}n^3$ flops

10.1.3 Implementation: Pseudoinverse via reduced SVD

1. Form the reduced SVD $A = \hat{U}\hat{\Sigma}V^*$ ($\sim 2mn^2 + 11n^3$ flops)
2. Compute $\vec{\mathbf{y}} = \hat{U}^*\vec{\mathbf{b}}$ ($\sim 2mn$ flops)
3. Compute $\vec{\mathbf{z}} = \hat{\Sigma}^{-1}\vec{\mathbf{y}}$ ($\sim n$ flops)
4. Compute $\vec{\mathbf{a}} = V\vec{\mathbf{z}}$ ($\sim 2n^2$ flops)

Total operation count: $\sim 2mn^2 + 11n^3$ flops

11 Normal Equations

Normal Equations

- A^*A is Hermitian
- If $rank(A^*A) = n$ it is positive definite since $x^*A^*Ax = \|Ax\|_2^2$ and $Ax = 0$ is only possible with $rank(A^*A) < n$.

12 Stability

12.1 Condition Numbers

Let $\vec{\mathbf{f}} : X \to Y$ be a continuous function. We say $\vec{\mathbf{f}}$ is well conditioned if small changes in $\vec{\mathbf{x}}$ result in small changes of $\vec{\mathbf{f}}(\vec{\mathbf{x}})$. That is if $\delta\vec{\mathbf{f}} = f(\vec{\mathbf{x}} + \delta\vec{\mathbf{x}}) - f(\vec{\mathbf{x}})$

- The **Absolute condition Number** is

$$\hat{\kappa} := \lim_{\delta\to 0} \sup_{\|\delta\vec{\mathbf{x}}\| \le \delta} \frac{\left\|\delta\vec{\mathbf{f}}\right\|}{\|\delta\vec{\mathbf{x}}\|}$$

  or if $\vec{\mathbf{f}}$ is differentiable, let $J(\vec{\mathbf{x}})$ be the Jacobian matrix of $\vec{\mathbf{f}}$ and so

$$\hat{\kappa} = \|J(\vec{\mathbf{x}})\|$$

- The **Relative Condition Number** is

$$\hat{\kappa} := \lim_{\delta\to 0} \sup_{\|\delta\vec{\mathbf{x}}\| \le \delta} \frac{\left\|\delta\vec{\mathbf{f}}\right\|}{\left\|\vec{\mathbf{f}}\right\|}\frac{\|\vec{\mathbf{x}}\|}{\|\delta\vec{\mathbf{x}}\|} = \frac{\|J(\vec{\mathbf{x}})\|\ \|\vec{\mathbf{x}}\|}{\left\|\vec{\mathbf{f}}\right\|}$$

- The **Condition Number of a Matrix**

$$\kappa(A) = \|A\|\ \left\|A^{-1}\right\|$$

  And if A is singular, $\kappa(A) = \infty$

Condition number facts

- If $A = A^*$ then one can show that the eigenvalues of $A + \delta A$ satisfy $|\delta\lambda| \le \|\delta A\|$.

NOTES: Jacobian Matrix

Theorem: Let $x \to f(x)$ be a problem with condition number $\kappa(x)$. Let $x \to \tilde{f}(x)$ be a backward stable algorithm. Then $\frac{\|\tilde{f}(x)-f(x)\|}{\|f(x)\|} = \kappa(x)O(()\epsilon_m)$ therefore a backward stable algorithm applied to a well conditioned problem is accurate.

Proof. $\tilde{f}$ backward stable implies $\tilde{f}(x) = f(\tilde{x})$ for some $\tilde{x}$ such that $\frac{\|x-\tilde{x}\|}{\|x\|} = O(()\epsilon_m)$.

$$\kappa(x) = \sup \frac{\|f(\tilde{x}) - f(x)\| \ \|x\|}{\|f(x)\| \ \|x - \tilde{x}\|}$$

So

$$\frac{\|f(\tilde{x}) - f(x)\|}{\|f(x)\|} \leq \kappa(x) \frac{\left\|x - \tilde{f}(x)\right\|}{\|x\|} \leq \kappa(x)O(()\epsilon_m)$$

Conditioning of QR: $A = QR$, $A + \delta A = (\hat{Q} + \delta Q)\hat{R}$, $\delta A = \delta\hat{Q}\hat{R}$, $\|\delta Q\| \ \leq \|\delta A\| \ \left\|\hat{R}^{-1}\right\|$

$$\frac{\|\delta Q\|_2 \|A\|_2}{\|\delta A\|_2 \left\|\hat{Q}\right\|_2} \leq \left\|\hat{R}^{-1}\right\|_2 \|A\|_2 = \kappa(A)$$

Householder in IEEE is backward stable. If $\tilde{Q}, \tilde{R}$ are the results of applying Householder to A, Then $\tilde{Q}$ is orthogonal, $\tilde{R}$ is upper triangular, and $\tilde{Q}\tilde{R} = A + \delta A$ where $\frac{\|\delta A\|}{\|A\|} = O(()\epsilon_m)$. $\tilde{Q}\tilde{R}$ is the exact QR factorization of $A + \delta A$.

Backsubstitution is backwards stable. Solution in IEEE satisfies $(R+\delta R)\tilde{x} = \vec{\mathbf{b}}$ for some upper triangular δR such that $\frac{\|\delta R\|}{\|R\|} = O(()\epsilon_m)$.

Example : Least Squares. Given $A \in \mathbb{C}^{m\times n}, m \geq n, \operatorname{rank}(A) = n$. $A\vec{\mathbf{x}} = \vec{\mathbf{b}}$,

$$\left\|\vec{\mathbf{b}} - A\hat{x}\right\|_2 = \min_{\vec{\mathbf{x}}} \left\|\vec{\mathbf{b}} - A\vec{\mathbf{x}}\right\|_2$$

$\hat{x} = f(A, \vec{\mathbf{b}})$, $\kappa(A, b)$ is complicated but roughly speaking $\kappa(A, b) \sim \kappa(A)$.

NOTES Vandermonde Matrix (in polynomial stuff)?

4th order compact finite difference. We can improve the accuracy of

$$y_i'' = D_+D_-y_i - \frac{h^2}{12}y_i^{(4)} + O(()h^4)$$

by using $y^{(4)}(x) = (q(x)y)'' - f''(x)$

$$y_i'' = D_+D_-y_i - \frac{h^2}{12}D_+D_-((qy)_i - f_i) + O(()h^4)$$

or

$$y_i'' = D_+D_-\left(\left(1 - \frac{h^2}{12}q_i\right)y_i\right) + \frac{h^2}{12}D_+D_-f_i + O(()h^4)$$

12.2 Gaussian Elimination

Reducing A to I via GE. For $A \in \mathbb{C}^{m\times m}$,

$$flops \sim \frac{4}{3}m^3$$

m columns, each column has $O(()m)$ entries, and takes $O(()m)$ multiplications and additions to zero out any given entry.

Direct methods for solving $A\vec{\mathbf{x}} = \vec{\mathbf{b}}$.

12.3 LU Facorization.

This is Gaussian Elimination where only adding multiples of rows to another are allowed

12.3.1 Goal

Find B such that $BA = U$ where U is upper triangular.

12.3.2 Process

B_1 eliminates all entries below a_{11} so B_1A has only zeros in the first column except a_{11}. Example: Assume for simplicity $a_{1,1} = 1$

$$B_1 = \begin{pmatrix} 1 & 0 & \\ -a_{2,1} & 1 & \\ -a_{3,1} & 0 & 1 \end{pmatrix}, \qquad B_2 = \begin{pmatrix} 1 & 0 & \\ & 1 & \\ & -a_{3,2} & 1 \end{pmatrix}$$

And so $B = B_n ... B_2 B_1$, so $B^{-1} = B_1^{-1} ... B_n^{-1}$, which turns out to be something like

$$\begin{pmatrix} 1 & 0 & \\ a_{2,1} & 1 & \\ a_{3,1} & a_{3,2} & 1 \end{pmatrix}$$

Then B_2 eliminates the subdiagonal entries of the second column of B_1A.

General LU. Let $\vec{\mathbf{x}}$ denote the Kth column of A at step K, B_k chosen so that $\vec{\mathbf{x}}_k = (x_{1,K}, x_{2,K}, ..., x_{k+1,K}, ..., x_{n,K})$. Then choose $l_{jk} = \frac{x_{jk}}{x_{kk}}$ then

$$L_{i,j} = \begin{cases} l_{i,j} & i > j \\ 1 & i = j \\ 0 & i < j \end{cases}$$

Algorithm in notes.

12.3.3 Problem

Existence of LU factorization fails for some nonsingular matrices. Even for those that exist, stability is not guaranteed.

12.3.4 Efficiency

LU factorization operation count:

$$flops \sim \sum_{k=1}^{m-1} \sum_{j=k+1}^{m} \sum_{s=k}^{m} 2 \sim \sum_{k=1}^{m-1} \sum_{j=k+1}^{m} 2(m-k) \sim \sum_{k=1}^{m-1} 2(m-k)^2 \sim 2\sum_{k=1}^{m-1} m^2 - 4m \sum_{k=1}^{m-1} k + 2\sum_{k=1}^{m-1} k^2 \sim 2m^3 - 2m^3 + \frac{2}{3}m^3 \sim \frac{2}{3}m^3$$

$2x$ more efficient than QR or computing A^{-1}. Forward subs $L\vec{\mathbf{x}} = b$ costs

$$flops \sim m^2$$

Which is negligible for large m compared to cost of LU factorization.
Backward subs $L\vec{\mathbf{x}} = b$ costs

$$flops \sim m^2$$

Which is negligible for large m compared to cost of LU factorization.

12.4 PLU factorization: $PA = LU$

Every nonsingular $A \in \mathbb{C}^{n\times n}$ has a PLU factorization.

12.5 Cholesky Factorization

LU decomposition for symmetric positive definite matrices.

Banded matrices Zero above a certain superdiagonal and/or zero below a certain subdiagonal.

- p is the number of superdiagonals
- q is the number of subdiagonals
- $0 \le p, q, \le m-1$
- The **Bandwidth** is $w = p+q+1$ so $1 \le w \le 2m-1$

Claim: If A has an LU decomposition then zeros are preserved. That is, if $A = LU$, then U has p superdiagonals and L has q superdiagonals and zero beyond.
For fixed p, q, and $m \to \infty$, the operation count of LU is $O(()ms^2) << O(()m^3)$ where $s = \max(p,q)$

If the matrix is sparse, A=sparse(m,m)

Sometimes you could have bands of zero within the bandwidth. This algorithm will fill in these zeros.

Claim: The PA=LU decomposition then the amount of non-zeros generally decreases. If A has bandwidth w, then L has bandwidth $w = q+1$ but U will have bandwidth $w = p+q+1$

Symmetric Positive Definite Matrices $A \in \mathbb{C}^{m\times m}$, $A = A^*$, and $\vec{\mathbf{x}}^* A\vec{\mathbf{x}} > 0$ for all $\vec{\mathbf{x}} \neq 0$.

Example $-y'' = f$ with second order central finite difference approximation.

$$A = (diag(-1,-1) + 2I + diag(-1,1))/h^2$$

We see that

$$\vec{\mathbf{x}}^* A\vec{\mathbf{x}} = \vec{\mathbf{x}}^* \begin{pmatrix} 2x_1 - x_2 \\ -x_1 + 2x_2 - x_3 \\ \vdots \\ -x_{m-1} + 2x_m \end{pmatrix} = 2x_1\bar{x}_1 - x_2\bar{x}_1 - x_1\bar{x}_2 + 2x_2\bar{x}_2 - \bar{x}_2 x_3 + \ldots$$

Note $|x_i - x_{i-1}|^2 = \bar{x}_i x_i - \bar{x}_{i-1}x_i - \bar{x}_i x_{i-1} + \bar{x}_{i-1}x_{i-1}$ thus

$$\vec{\mathbf{x}}^* A\vec{\mathbf{x}} = |x_1|^2 + |x_m|^2 + \sum_{i=2}^{m} |x_i - x_{i-1}|^2$$

So $\vec{\mathbf{x}}^* A\vec{\mathbf{x}} \ge 0$ but if $\vec{\mathbf{x}}^* A\vec{\mathbf{x}} = 0$, then

$$|x_1| = 0 \implies x_1 = 0 \implies x_2 = 0 \implies .... \implies x_m = 0$$

Example $-\nabla\phi = f$. So $-\phi_{,x,x} + \phi_{,y,y} = f$ on $D \subset \mathbb{R}^2$. Using Lexicographic ordering,

$$4 - 10 - 14 - 10 - 1 - 1004 - 10 - 1 - 10 - 14 - 10 - 1(u_1, u_2, ..., u_9)^T = (h^2 f_{11}, f_2 f_{21}, ..., f_{33})^T + BC$$

A is block tridiagonal. A is symmetric and positive definite. A is positive definite if and only if eigenvalues of A are all positive. The N^2 eigenvalues of

$$\lambda^{p,q} = \frac{2}{h^2}\left((1 - \cos(p\pi h)) + (1 - \cos(q\pi h))\right)$$

So $\lambda^{p,q} > 0$ for $p, q = 1, 2, ..., N$

Cholesky Factorization (modified LU for SPD(symmetric positive definite)) Let $A \in \mathbb{C}^{n\times n}$ be SPD

$$A = \begin{pmatrix} a_{11} & \vec{\mathbf{w}}^* \\ \vec{\mathbf{w}} & B \end{pmatrix}$$

Where $B \in \mathbb{C}^{(m-1)\times(m-1)}$ is SPD and $\vec{\mathbf{w}} \in \mathbb{C}^{(m-1)}$. $a_11 = \vec{\mathbf{e}}_1^* A \vec{\mathbf{e}}_1 > 0$. We perform the first step of Gaussian elimination

$$\begin{aligned} A &= \begin{pmatrix} 1 & \vec{\mathbf{0}}^* \\ a_{11}^{-1}\vec{\mathbf{w}} & I \end{pmatrix} \begin{pmatrix} a_{11} & \vec{\mathbf{w}}^* \\ \vec{\mathbf{0}} & B - \frac{\vec{\mathbf{w}}\vec{\mathbf{w}}^*}{a_{11}} \end{pmatrix} \\ &= \begin{pmatrix} 1 & \vec{\mathbf{0}}^* \\ a_{11}^{-1}\vec{\mathbf{w}} & I \end{pmatrix} \begin{pmatrix} a_{11} & \vec{\mathbf{0}}^* \\ \vec{\mathbf{0}} & B - \frac{\vec{\mathbf{w}}\vec{\mathbf{w}}^*}{a_{11}} \end{pmatrix} \begin{pmatrix} 1 & a_{11}^{-1}\vec{\mathbf{w}}^* \\ \vec{\mathbf{0}} & I \end{pmatrix} \\ &= \begin{pmatrix} \sqrt{a_{11}} & \vec{\mathbf{0}}^* \\ \sqrt{a_{11}}^{-1}\vec{\mathbf{w}} & I \end{pmatrix} \begin{pmatrix} 1 & \vec{\mathbf{0}}^* \\ \vec{\mathbf{0}} & B - \frac{\vec{\mathbf{w}}\vec{\mathbf{w}}^*}{a_{11}} \end{pmatrix} \begin{pmatrix} \sqrt{a_{11}} & \sqrt{a_{11}}^{-1}\vec{\mathbf{w}}^* \\ \vec{\mathbf{0}} & I \end{pmatrix} \\ &= R_1^* A_2 R_1 \end{aligned}$$

Properties

- Every SPD $A \in \mathbb{C}^{n\times n}$ has a unique Cholesky Factorization.
- The Bandwidth is preserved. If A has bandwidth $w = 2p+1$, then R has bandwidth $p+1$.
- No need for pivoting.
- Algorithm is backwards stable for $Ax = b$. $\tilde{R}^*\tilde{R} = A + \delta A$ where $\frac{\|\delta A\|}{\|A\|} = O(()\epsilon_{machine})$ for some $\delta A \in \mathbb{C}^{n\times n}$. Cholesky gives us $\tilde{x}$ that satisfies $(A + \delta A)\tilde{x} = b$.

Usage:
To solve $Ax = b$, obtain $A = R^*R$. Then solve $R^*\vec{\mathbf{y}} = b$ via Forward subs, and $R\vec{\mathbf{x}} = \vec{\mathbf{y}}$ via Backward subs.
Claim: $A_2 = (R_1^*)^{-1}AR_1^{-1}$ is SPD thus $B - \frac{\vec{\mathbf{w}}\vec{\mathbf{w}}^*}{a_{11}}$ is SPD. Proof:

- Symmetric:
$$A_2^* = (R_1^*)^{-1}AR_1^{-1} = A_2$$
- Positive Definite
$$\vec{\mathbf{x}}^*A_2\vec{\mathbf{x}} = \vec{\mathbf{x}}^*(R_1^*)^{-1}AR_1^{-1}\vec{\mathbf{x}} = (R_1^{-1}\vec{\mathbf{x}})A(R_1^{-1}\vec{\mathbf{x}}) = \vec{\mathbf{y}}^*A\vec{\mathbf{y}} > 0$$
- $B - \frac{\vec{\mathbf{w}}\vec{\mathbf{w}}^*}{a_{11}}$ is SPD. We repeat the Cholesky step m times
$$A = R_1^*A_2R_1 = R_1^*R_2^*A_3R_2R_1 = ... = (R_1^*R_2^*...R_m^*)I(R_m...R_1) = R^*IR = R^*R$$

Operation Count:

$$flops \sim \sum_{k=1}^{m}\sum_{j=(k+1)}^{m}\sum_{s=j}^{m} 2 = \sum_{k=1}^{m}\sum_{j=(k+1)}^{m} 2(m-j+1) = \sum_{k=1}^{m}\sum_{n=1}^{m-k} 2n = \sum_{k=1}^{m}(m-k)^2 = m^3 - 2m\left(\frac{1}{2}m^2\right) + \frac{1}{3}m^3 = \frac{1}{3}m^3$$

Thus this is twice as efficient as LU.

13 Numerical Eigenvalue Problems

Let $A \in \mathbb{C}^{m \times m}$. The eigenvalue problem is

$$A\vec{\mathbf{x}} = \lambda\vec{\mathbf{x}}, \qquad \vec{\mathbf{x}} \neq 0$$

MISSING NOTES THURS 10

- Phase 1: Reduce A to upper Hessenberg form via unitary similarity transformations.
  $A \in \mathbb{C}^{m \times m}$ implies in $2m-1$ steps we can get our H matrix

$$Q_{m-2}Q_{m-3}...Q_2Q_1AQ_1Q_2...Q_{m-3}Q_{m-2} = Q^*AQ = H$$

- Phase 2: Iterate to reveal eigenvalues of H.

Operation count: 1st inner loop: $\sum\limits_{k=1}^{m-2} 4(m-k)^2 \sim \frac{4}{3}m^3$
2nd inner loop: $\sum\limits_{k=1}^{m-2} 4m(m-k) \sim 4m\frac{m^2}{2} = 2m^3$
Total= $\frac{10}{3}m^3$

The algorithm is backwards stable with conditioning cond(A)
If A is Hermitian, then H is Hermitian.

$$A - A^* = QHQ^* - QH^*Q^* = Q(H - H^*)Q^* = 0$$

Therefore H is tridiagonal. In this case, a simplified algorithm is possible, reducing the operation count to $\frac{4}{3}m^3$.
Phase 2 For remainder of discussion, assume $A \in \mathbb{R}^{n \times n}$ and $A^T = A$, thus $A = QDQ^T$.
$\{\lambda_i\}$ are real, and eigenvectors $\{\vec{\mathbf{q}}_i\}$ are orthonormal.
Rayleigh Quotient Given $\vec{\mathbf{x}} \in \mathbb{R}^m$, find a value $\hat{\alpha} \in \mathbb{R}$ such that

$$\|A\vec{\mathbf{x}} - \hat{\alpha}\vec{\mathbf{x}}\|_2 = \min_{\alpha} \|A\vec{\mathbf{x}} - \alpha\vec{\mathbf{x}}\|_2$$

$$\|A\vec{\mathbf{x}} - \hat{\alpha}\vec{\mathbf{x}}\|_2 = (A\vec{\mathbf{x}} - \hat{\alpha}\vec{\mathbf{x}})^T(A\vec{\mathbf{x}} - \hat{\alpha}\vec{\mathbf{x}}) = \vec{\mathbf{x}}^TA^2\vec{\mathbf{x}} - 2\hat{\alpha}\vec{\mathbf{x}}^TA\vec{\mathbf{x}} + \hat{\alpha}^2\vec{\mathbf{x}}^*\vec{\mathbf{x}}$$

Taking the derivative with respect to α and setting to zero we get

$$-2\vec{\mathbf{x}}^TA\vec{\mathbf{x}} + 2\hat{\alpha}\vec{\mathbf{x}}^T\vec{\mathbf{x}} = 0 \implies \hat{\alpha} = \frac{\vec{\mathbf{x}}^TA\vec{\mathbf{x}}}{\vec{\mathbf{x}}^T\vec{\mathbf{x}}} = R_A(\vec{\mathbf{x}})$$

(Rayleigh Quotient)
Error Estimate: (Taylor Series)

$$R_A(\vec{\mathbf{x}}) = R_A(q_j) + \nabla R_A(q_j) \cdot (\vec{\mathbf{x}} - \vec{\mathbf{q}}_j) + O(()) \|\vec{\mathbf{x}} - \vec{\mathbf{q}}_j\|^2$$

$$\nabla R_A(\vec{\mathbf{x}}) = \nabla\left(\frac{\vec{\mathbf{x}}^TA\vec{\mathbf{x}}}{\vec{\mathbf{x}}^T\vec{\mathbf{x}}}\right) = \frac{\vec{\mathbf{x}}^T\vec{\mathbf{x}}\nabla(\vec{\mathbf{x}}^TA\vec{\mathbf{x}}) - (\vec{\mathbf{x}}^TA\vec{\mathbf{x}})\nabla(\vec{\mathbf{x}}^T\vec{\mathbf{x}})}{(\vec{\mathbf{x}}^T\vec{\mathbf{x}})^2}$$

Since $\nabla(\vec{\mathbf{x}}^T\vec{\mathbf{x}}) = \nabla(x_1^2 + ... + x_m^2) = 2\vec{\mathbf{x}}^T$ and $\nabla(\vec{\mathbf{x}}^TA\vec{\mathbf{x}}) = 2(A\vec{\mathbf{x}})^T$

$$\nabla R_A(\vec{\mathbf{x}}) = \frac{2}{\vec{\mathbf{x}}^T\vec{\mathbf{x}}}\left((A\vec{\mathbf{x}})^T - R_A(\vec{\mathbf{x}})\vec{\mathbf{x}}^T\right)$$

Thus for an orthonormal eigenvctor q_i,

$$\nabla R_A(\vec{\mathbf{q}}_i) = 2\left(\lambda_i \vec{\mathbf{q}}_i - \lambda_i \vec{\mathbf{q}}_i^T\right) = 0$$

So we conclude

$$R_A(\vec{\mathbf{x}}) = \lambda_j + O(())\left\|\vec{\mathbf{x}} - \vec{\mathbf{q}}_j\right\|^2$$

Method 1: Power method Assume $|\lambda_1| > |\lambda_2| \geq ... \geq |\lambda_m|$. Start with vector $\vec{\mathbf{w}}^{(0)} \in \mathbb{R}^m$.

$$w^{(0)} = \alpha_1 \vec{\mathbf{q}}_1 + ... + \alpha_m \vec{\mathbf{q}}_m$$

$$Aw^{(0)} = \alpha_1 A\vec{\mathbf{q}}_1 + ... + \alpha_m A\vec{\mathbf{q}}_m = \alpha_1 \lambda_1 \vec{\mathbf{q}}_1 + ... + \alpha_m \lambda_m \vec{\mathbf{q}}_m$$

Assume this is $\approx \alpha_1 \lambda_1 \vec{\mathbf{q}}_1$ (that $\lambda_1 >> \lambda_i, i > 1$). Improved guess is $\vec{\mathbf{w}}^{(1)} = A\vec{\mathbf{w}}^{(0)}$. Iterating, $\vec{\mathbf{w}}^{(n)} = A\vec{\mathbf{w}}^{n-1)}$

$$\vec{\mathbf{w}}^{(n)} = \alpha_1 A^n \vec{\mathbf{q}}_1 + ... + \alpha_m A^n \vec{\mathbf{q}}_m = \alpha_1 \lambda_1^n \vec{\mathbf{q}}_1 + ... + \alpha_m \lambda_m^n \vec{\mathbf{q}}_m \approx \alpha_1 \lambda_1^n \vec{\mathbf{q}}_1$$

Theorem: Suppose $|\lambda_1| > |\lambda_2| \geq ... \geq |\lambda_m|$ and $q_1^T v^{(0)} \neq 0$. The power iteration after k steps produces an approximate eigenvector eigenvalue pair that satisfies

$$\left\|\vec{\mathbf{v}}^{(k)} - \pm\vec{\mathbf{q}}_1\right\|_2 = O(())\left|\frac{\lambda_2}{\lambda_1}\right|^k, \qquad \left|\lambda^{(k)} - \lambda_1\right| = O(())\left|\frac{\lambda_2}{\lambda_1}\right|^{2k}$$

Proof. $\vec{\mathbf{v}}^{(0)} = \alpha_1 \vec{\mathbf{q}}_1 + ... + \alpha_m \vec{\mathbf{q}}_m$.

$$\vec{\mathbf{v}}^{(k)} = \beta_k A^k \vec{\mathbf{v}}^0 = \beta_k A^k\left(\alpha_1 \vec{\mathbf{q}}_1 + ... + \alpha_m \vec{\mathbf{q}}_m\right) = \beta_k \alpha^1 \lambda_k^1\left(\vec{\mathbf{q}}_1 + ... + \frac{\alpha_m}{\alpha_1}\left(\frac{m}{\lambda_1}\right)^k \vec{\mathbf{q}}_m\right) \to \beta_k \alpha^1 \lambda_k^1$$

Thus $\left\|\vec{\mathbf{v}}^{(k)} - (\pm q_1)\right\| = O(())\left|\frac{\lambda_2}{\lambda_1}\right|^k$. Also

$$\lambda^{(k)} = R_A(\vec{\mathbf{v}}^{(k)} = R_A(\vec{\mathbf{q}}_i) + O(())\|\vec{}(k) - (\pm\vec{\mathbf{q}}_1)\|^2) = \lambda_1 + O(\left|\frac{\lambda_2}{\lambda_1}\right|^{2k})$$

Limitations of Power Iteration

-

Only gives largest eigenvalue λ_1

$\vec{\mathbf{v}}^{(k)}, \lambda^{(k)}$ converge only linearly:

$$\lim_{n\to\infty} \frac{\|\vec{}(k+1) - (\pm\vec{\mathbf{q}}_1)\|}{\|\vec{}(k) - (\pm\vec{\mathbf{q}}_1)\|} = c\left|\frac{\lambda_2}{\lambda_1}\right|, 0 < c < 1$$

which is very slow if $\frac{\lambda_2}{\lambda_1}$ is close to 1.

13.1 Shifted Inverse Iteration

Applying the Power Iteration Method to A^{-1} yields an approximation to $\frac{1}{|\lambda_m|}$ and $\vec{\mathbf{q}}_m$.
Applying the Power Iteration Method to $(A - \mu I)^{-1}$, which has eigenvalues $(\lambda_j - \mu)^{-1}$ and eigenvectors q_j, yields approximation to $(\lambda_k - \mu)^{-1}$ and q_k where $|\lambda_k - \mu| = \min_j |\lambda_j - \mu|$.

Suppose λ_J is the eigenvalue closest to μ and λ_K is the next closest.

$$|\lambda_J - \mu| < |\lambda_K - \mu| \leq |\lambda_i - \mu| \ \forall\, i \neq j$$

Assume $q_J^T \vec{\mathbf{v}}^{(0)} \neq 0$. The shifted inverse iteration produces approximations that satisfy $\left\|\vec{\mathbf{v}}^{(k)} - (\pm\vec{\mathbf{q}}_J)\right\| = O(\left(\left|\frac{\lambda_J - \mu}{\lambda_K - \mu}\right|^K\right))$ $\left\|\lambda^{(k)} - \lambda_J\right\| = O(\left(\left|\frac{\lambda_J - \mu}{\lambda_K - \mu}\right|^{2K}\right))$

Proof. Proof: same as power iteration $\lambda_1 \to \frac{1}{\lambda_J - \mu}$, $\lambda_2 \to \frac{1}{\lambda_K - \mu}$ We still have linear convergence, but the constant c depends on μ so we may have faster convergence.

Note. Accuracy of the linear solver of $(A - \mu I)\vec{\mathbf{w}} = \vec{\mathbf{v}}$ depends on $\kappa(A - \mu I)$ but for good guesses of μ this could be huge. However, if this is solved by a backwards stable algorithm, then $\|\vec{\mathbf{w}} - \tilde{w}\| = \kappa(A - \mu I)O(()\epsilon_{machine})$. Therefore if μ is close to λ_J then error $\|\vec{\mathbf{w}} - \tilde{w}\|$ could be large. However, even if this is the vase, $\vec{\mathbf{v}}^{(k)} = \frac{\vec{\mathbf{w}}}{\|\tilde{w}\|}$ is still a good approximation to $\vec{\mathbf{q}}_J$ since we only need the direction.

Proof. $(A - \mu I)\vec{\mathbf{w}} = \vec{\mathbf{v}}$ and $(A - \mu I)\tilde{w} = \tilde{v}$, so

$$(A - \mu I)(\vec{\mathbf{w}} - \tilde{w}) = \vec{\mathbf{v}} - \tilde{v} = \alpha_1 \vec{\mathbf{q}}_1 + ... + \alpha_m \vec{\mathbf{q}}_m$$

$$\vec{\mathbf{w}} - \tilde{w} = \alpha_1 (A - \mu I)^{-1}(\vec{\mathbf{v}} - \tilde{v}) = \alpha_1 (\lambda_1 - \mu)^{-1}\vec{\mathbf{q}}_1 + ... + \alpha_m(\lambda_1 - \mu)^{-1}\vec{\mathbf{q}}_m$$

Therefore $\vec{\mathbf{w}} - \tilde{w} \approx \alpha_J(\lambda_J - \mu)^{-1}\vec{\mathbf{q}}_J$. The bulk of the error in $\vec{\mathbf{w}}$ is in the direction of $\vec{\mathbf{q}}_J$ so $\vec{\mathbf{v}}^{(k)} = \frac{\tilde{w}}{\|\tilde{w}\|_2}$ is a good approximation to $\vec{\mathbf{q}}_J$.

Theorem If $\vec{\mathbf{v}}^{(0)} \in \mathbb{R}^m$ is sufficiently close to the eigenvector $\vec{\mathbf{q}}_J \in \mathbb{R}^m$ then Rayleigh Quotient Iteration produces approximations that satisfy $\left\|\vec{\mathbf{v}}^{(k)} - (\pm\vec{\mathbf{q}}_J)\right\| = O(\left\|\vec{\mathbf{v}}^{(k)} - (\pm\vec{\mathbf{q}}_J)\right\|^3)$ $\left|\lambda^{(k+1)}\right| = O(()\left|\lambda^{(k)} - \lambda_J\right|^3)$

Proof. Proof: $(A - \lambda^{(k)}I)\vec{\mathbf{w}} = \vec{\mathbf{v}}^{(k)}$

$$\begin{aligned}
\vec{\mathbf{w}} =&(A - \lambda^{(k)}I)^{-1}\vec{\mathbf{v}}^{(k)} \\
=&(A - \lambda^{(k)}I)^{-1}(\vec{\mathbf{q}}_J + \vec{\mathbf{v}}^{(k)} - \vec{\mathbf{q}}_J) \\
&(A - \lambda^{(k)}I)^{-1}\vec{\mathbf{q}}_J = \frac{1}{\lambda_J - \lambda^{(k)}}\vec{\mathbf{q}}_J \\
&\vec{\mathbf{v}}^{(k)} - \vec{\mathbf{q}}_J \approx \vec{\mathbf{q}}_K \\
&(A - \lambda^{(k)}I)^{-1}\vec{\mathbf{v}}^{(k)} - \vec{\mathbf{q}}_J \approx \frac{1}{\lambda_K - \lambda^{(k)}}(\vec{\mathbf{v}}^{(k)} - \vec{\mathbf{q}}_J) \approx \frac{1}{\lambda_K - \lambda^{(k)}}(\vec{\mathbf{v}}^{(k)} - \vec{\mathbf{q}}_J)
\end{aligned}$$

$$\vec{\mathbf{w}} \approx \frac{1}{\lambda_J - \lambda^{(k)}}\vec{\mathbf{q}}_J + \frac{1}{\lambda_K - \lambda^{(k)}}(\vec{\mathbf{v}}^{(k)} - \vec{\mathbf{q}}_J)$$

$$\vec{\mathbf{v}}^{(k+1)} = \frac{\vec{\mathbf{w}}}{\|\vec{\mathbf{w}}\|_2} \approx \vec{\mathbf{q}}_J + \frac{\lambda_J - \lambda^{(k)}}{\lambda_K - \lambda^{(k)}}(\vec{\mathbf{v}}^{(k)} - \vec{\mathbf{q}}_J)$$

$$\begin{aligned}
(\vec{\mathbf{v}}^{(k+1)} - \vec{\mathbf{q}}_J) \approx& \frac{\lambda_J - \lambda^{(k)}}{\lambda_K - \lambda^{(k)}}(\vec{\mathbf{v}}^{(k)} - \vec{\mathbf{q}}_J) \\
=&O(()\left|\lambda_J - \lambda^{(k)}\right| \|\vec{}(k) - \vec{\mathbf{q}}_J\|)
\end{aligned}$$

but $\left|\lambda_J - \lambda^{(k)}\right| = O(()\|\vec{}(k) - \vec{\mathbf{q}}_J\|^2)$ from RQ Taylor series, which implies

$$\left\|\vec{\mathbf{v}}^{(k+1)} - \vec{\mathbf{q}}_J\right\| = O(()\|\vec{}(k) - \vec{\mathbf{q}}_J\|^3)$$

Thus $\left|\lambda^{(k+1)} - \lambda_J\right| = O(()\left\|\vec{\mathbf{v}}^{k+1} - \vec{\mathbf{q}}_J\right\|^2) = O(()\left\|\vec{\mathbf{v}}^{k} - \vec{\mathbf{q}}_J\right\|^6) = O(()\left|\lambda^{(k) - \lambda_J}\right|)$ Operation count (cost of 1 iteration)

	$FullMatrix$	$Tridiagonal$
Power	$O(()m^2)$(Matrix Vector Multiply)	$O(()m)$
Inverse	$O(()m^2)$(Precomputed Cholesky)	$O(()m)$
Rayleigh	$O(()m^3)$(Cholesky each time)	$O(()m)$

Therefore we can use the phase 1 reduction (which is $O(()m^3)$ to improve efficiency)
Note: Algorithms so far allow us to compute only a single eigenvalue at a time.

13.1.1 QR Algorithm

$$A^{(k)} = R^{(k)}Q^{(k)}, \qquad R^{(k)} = (Q^{(k)})^T A^{(k-1)}$$
$$A^{(k)} = (Q^{(k)})^T A^{(k-1)} (Q^{(k)})$$

This is a similarity transformation, so it will have the same eigenvalues. As $k \to \infty$,

$$A^{(k)} \to \begin{cases} D & A \text{ is normal} \\ T & \text{otherwise} \end{cases}$$

Theorem: Assume that $A \in \mathbb{R}^{n\times n}$ is symmectric such that $A = Q\Lambda Q^T$ is the spectral decomposition where $|\lambda_1| > |\lambda_2| > ... > |\lambda_m| > 0$ so $\det(\Delta_k(Q)) \neq 0$ for $k = 1, 2, ...m$. Then $A^{(k)} \to D$ as $k \to \infty$.

Proof. Proof:

$$A^{(k)} = \bar{Q}^{(k)} A (\bar{Q}^{(k)})^T$$

, $\quad \bar{Q}^{(k)} = Q^{(1)}...Q^{(k)}$Therefore we need to prove that $\bar{Q}^{(k)} \to Q$ as $k \to \infty$ because then $A^{(k)} \to Q^T A Q = D$.

$$A = A^{(0)} = Q^{(1)}R^{(1)}, A^2 = Q^{(1)}R^{(1)}Q^{(1)}R^{(1)} = Q^{(1)}A^{(1)}R^{(1)}$$

since $R^{(1)}Q^{(1)} = A^{(1)} = Q^{(2)}R^{(2)}$.

$$A^2 = Q^{(1)}Q^{(2)}R^{(2)}R^{(1)}, \qquad A^3 = Q^{(1)}Q^{(2)}Q^{(3)}R^{(3)}R^{(2)}R^{(1)}$$

so by the above definitions, $A^k = \bar{(}Q^{(k)})^T \bar{R}^{(k)}$ but

$$A^k = \left[A^k\vec{\mathbf{e}}_1, ..., A^k\vec{\mathbf{e}}_m\right]$$

This simulates power iteration. Let $\vec{\mathbf{e}}_1 = \alpha_{11}\vec{\mathbf{q}}_1 + ... + \alpha_{1m}\vec{\mathbf{q}}_m$ where $\{q_i\}$ are the columns of Q. Take the dot product of this with $\vec{\mathbf{q}}_1$, $\vec{\mathbf{q}}_1^T\vec{\mathbf{e}}_1 = \alpha_{11} = \vec{\mathbf{e}}_1^T\vec{\mathbf{q}}_1 = det(\Delta_1(Q)) \neq 0$ Therefore $\alpha_{11} \neq 0$. So

$$\frac{1}{\alpha_{11}}\vec{\mathbf{e}}_1 = \vec{\mathbf{q}}_1 + ... + \frac{\alpha_{1m}}{\alpha_{11}}\vec{\mathbf{q}}_m$$

Multiplying by A^k,

$$\frac{1}{\alpha_{11}}A^k\vec{\mathbf{e}}_1 = \lambda_1^k\vec{\mathbf{q}}_1 + ... + \frac{\alpha_{1m}}{\alpha_{11}}\lambda_m^k\vec{\mathbf{q}}_m$$

This becomes

$$\frac{1}{\alpha_{11}}\frac{\bar{r}_{11}^{(k)}}{\lambda_1^k}\vec{\mathbf{q}}_1^{(k)} = \vec{\mathbf{q}}_1 + ... + \frac{\lambda_m^k}{\lambda_1^k} frac\alpha_{1m}\alpha_{11}\vec{\mathbf{q}}_m$$

So $\vec{\mathbf{q}}_1^{(k)} \to \vec{\mathbf{q}}_1$ as $k \to \infty$. Now,

$$\vec{\mathbf{e}}_2 = \alpha_{21}\left(\frac{1}{\alpha_{11}}\vec{\mathbf{e}}_1 - \left(\frac{\alpha_{12}}{\alpha_{11}}\vec{\mathbf{q}}_2 + ... + \frac{\alpha_{1m}}{\alpha_{11}}\vec{\mathbf{q}}_m\right)\right) + \alpha_{22}\vec{\mathbf{q}}_2 + ...$$

$$\vec{\mathbf{e}}_2 = \frac{\alpha_{22}}{\alpha_{11}}\vec{\mathbf{e}}_1 + \frac{\alpha_{11}\alpha_{22} - \alpha_{12}\alpha_{21}}{\alpha_{11}}\vec{\mathbf{q}}_2 + \tilde{\alpha}_{23}\vec{\mathbf{q}}_3 + ... + \tilde{\alpha}_{2m}\vec{\mathbf{q}} + ... + \tilde{\alpha}_{2m}\vec{\mathbf{q_m}}$$

$$\frac{\alpha_{11}\alpha_{22} - \alpha_{12}\alpha_{21}}{\alpha_{11}} = \frac{1}{\alpha_{11}}\det\left(\begin{pmatrix} \alpha_{11} & \alpha_{12} \\ \alpha_{21} & \alpha_{22} \end{pmatrix}\right) = \frac{1}{\alpha_{11}}\det\left(\begin{pmatrix} \vec{\mathbf{q}}_1^T\vec{\mathbf{e}}_1 & \vec{\mathbf{q}}_2^T\vec{\mathbf{e}}_1 \\ \vec{\mathbf{q}}_1^T\vec{\mathbf{e}}_2 & \vec{\mathbf{q}}_2^T\vec{\mathbf{e}}_2 \end{pmatrix}\right) = \det(\Delta_2(Q)) \neq 0$$

Multiplying by A^K, etc. as before gives us that $Q^{(k)} \to Q$ as $k \to \infty$.
One can procede inductively.

Note

- $A^{(0)}$ is tridiagonal from Phase 1.
- $A^{(k)}$ is also tridiagonal for any finite k.
- As $k \to \infty$, off diagonal elements converge to 0.

3. Deflation: If $|A_{j,j+1}| < \epsilon$, then set $A^{(k)} = \begin{pmatrix} A_1 & \\ & A_2 \end{pmatrix}$. $A_1 \in \mathbb{R}^{j\times j}$, $A_2 \in \mathbb{R}^{(m-j)\times(m-j)}$. Then continue iterating on A_1 and A_2 independently.
4. Accelerate by shifting

$$Q^{(k)}R^{(k)} = A^{(k-1)} - \mu^{(k)}I$$

Where $\mu^{(k)}$ is usually chosen to be $A^{(k)}_{mm}$ To recreate A, use

$$= A^{(k-1)} = Q^{(k)}R^{(k)} + \mu^{(k)}I$$

13.2 Iterative Schemes for $Ax = b$

Given an interative scheme where $A = M + N$,

$$M\vec{\mathbf{x}}^{(k+1)} = N\vec{\mathbf{x}}^{(k)} + b$$

Thus if we have the amplification matrix $G = M^{-1}N$,

$$\vec{\mathbf{e}}^{(k+1)} = G\vec{\mathbf{e}}^{(k)}$$

By induction, $\vec{\mathbf{e}}^{(k+1)} = G^k\vec{\mathbf{e}}^{(0)}$. Perform the spectral decomposition of G,

$$\vec{\mathbf{e}}^{(k)} = R\Gamma^K R^{-1}\vec{\mathbf{e}}^{(0)}$$

If $\Gamma = \mathrm{diag}(\gamma_1, ..., \gamma_n)$ with $|\gamma_1| \geq ... \geq |\gamma_m|$ then we have convergence if $|\gamma_1| < 1.$ since then $G^K \to 0$ as $k \to \infty$.

13.2.1 Jacobi Iteration

$$M = D, N = D - A \implies G = I - D^{-1}A$$

Say we are considering the discrete Laplacian operator problem $-\Delta u = f$. For this situation, $G = I - \frac{h^2}{2}A$ so A and G have the same eigenvectors and the eigenvalues are related by

$$\gamma_p = 1 - \frac{h^2}{2}\lambda_p = 1 - \frac{h^2}{2}\frac{2}{h^2}(1 - \cos(p\pi h) = \cos(p\pi h)$$

$\gamma_1 = \cos(\pi h) = 1 - \frac{1}{2}\pi^2h^2 + \emptyset(h^4)$. Therefore $\rho(G) < 1 \,\forall\, h > 0$ but $\rho(G) \to 1$ as $h \to 0$. Since this converges quadratically at $h \to 0$, we expect slow convergence for the scheme.
How many iterations will we need?

$$\left\|\vec{\mathbf{e}}^{(k)}\right\| \approx \epsilon\left\|\vec{\mathbf{e}}^{(0)}\right\| \implies \rho^k = \epsilon \implies k \approx \frac{\log(\epsilon)}{\log(\rho)}$$

How do we choose ϵ? Our error for $A\vec{\mathbf{u}} = \vec{\mathbf{f}}$ is $O(()h^2)$, so take $\epsilon \approx ch^2$

$$k = \frac{\log(c) + 2\log(h)}{\log(\rho)} \approx \frac{\log(c) + 2\log(m+1)}{-\frac{1}{2}\pi^2(m+1)^2} \sim \frac{4}{\pi^2}m^2\log(m) = O(()m^2\log(m))$$

The total work is $O(()m^3\log(m))$.
A similar analysis can be performed for $m = N^d$ where N is the number of grid points in each coordinate direction. The total work turns out to be $O(()N^{d+2}\log(N))$.

13.2.2 Gauss Seidel

$$M = L + D, N = U \implies G = (L+D)^{-1}R$$

$\rho(G) = 1 - \pi^2 h^2 + O(()h^4)$ as $h \to 0$. Therefore this takes half as many iterations as Jacobi. Still ttoal work $O(()m^3 \log(m)$.

13.2.3 Successive Over-Relaxation (SOR)

Idea:$\vec{\mathbf{u}}^{(k+1)}$ is closer to $\vec{\mathbf{u}}$ than $\vec{\mathbf{u}}^{(k)}$ is to $\vec{\mathbf{u}}$, but not by much. That is, GS moves $\vec{\mathbf{u}}^{(k)}$ in the right direction but not far enough.

$$A = D + L + U, M = \frac{1}{\omega}(D - \omega L), N = \frac{1}{\omega}((1-\omega)D + \omega U)$$

Theorem: Ostrowski. If $A \in \mathbb{R}^{n \times n}$ is SPD, and $D - \omega L$ is nonsingular, then SOR converges for all $0 < \omega < 2$

For our standard $u'' = f$ BVP,

$$\text{(Stage 1)} u_i^{GS} = \frac{1}{2}(\vec{\mathbf{u}}^{(k+1)} + \vec{\mathbf{u}}^{(k)} + h^2 f_i)$$

$$\text{(Stage 2)} u_i^{(k+1)} = u_i^{(k)} + \omega(\vec{\mathbf{u}}_i^{GS} - \vec{\mathbf{u}}^{(k)})$$

Think of $\vec{\mathbf{u}}_i^{GS} - \vec{\mathbf{u}}^{(k)}$ as the direction and ω is the step size, starting from $u_i^{(k)}$.
For our model problem,

$$\omega_{optimal} = \frac{2}{1 + \sin(\pi h)} \approx 2 - 2\pi h \implies \rho_{optimal} = \omega_{optimal} - 1 \approx 1 - 2\pi h$$

So $\rho_{optimal} \to 1$ as $h \to 0$, but much more slowly than GS or Jacobi. $K_{optimal} = O(()N \log(N))$ in any dimension. The total work is $O(()N^{d+1} \log(N))$.

13.3 Descent Methods

Consider $\phi : \mathbb{R}^m \to \mathbb{R}$ defined by

$$\phi(\vec{\mathbf{u}}) = \frac{1}{2}\vec{\mathbf{u}}^T A \vec{\mathbf{u}} - b^T \vec{\mathbf{u}}$$

where $A \in \mathbb{R}^{n \times n}$ is SPD. Solve $A\vec{\mathbf{u}} = b$
Claim: ϕ has a unique global minimizer, $\vec{\mathbf{u}}^*$, that satisfies $A\vec{\mathbf{u}} =^* b$.
Proof: ϕ is quadratic in $u_1, ..., u_m$. Thus $\lim_{\|\vec{\mathbf{u}}\| \to \infty} \phi(\vec{\mathbf{u}}) = +\infty$ because $\vec{\mathbf{u}}^T A \vec{\mathbf{u}} > 0$.

$$\phi(\vec{\mathbf{u}}) = \frac{1}{2}\sum_{i=1}^m u_i \left(\sum_{j=1}^m a_{ij} u_j\right) - \sum_{i=1}^m b_i u_i$$

$$\frac{\partial \phi}{\partial u_k} = \frac{1}{2}\left(\sum_{j=1}^m a_{kj} u_j + \sum_{i=1}^m u_i a_{ik}\right) - b_k = 0$$

But since $a_{ik} = a_{ki}$,

$$\frac{\partial \phi}{\partial u_k} = \left(\sum_{j=1}^m a_{kj} u_j\right) - b_k = (A\vec{\mathbf{u}} - \vec{\mathbf{b}})_k = 0$$

Therefore $\vec{\mathbf{u}}^*$ such that $A\vec{\mathbf{u}}^* = b$ is the only critical point. By positive definiteness, it is the unique global minimizer.

So our strategy will be to (starting with a guess $\vec{\mathbf{u}}^{(0)}$) form the sequence $\left\{\vec{\mathbf{u}}^{(k)}\right\}_{k=0}$ by chooseing a search direction $\vec{\mathbf{d}}^{(k)}$ and a step size $\alpha^{(k)}$ such that

$$\vec{\mathbf{u}}^{(k+1)} = \vec{u}^{(k)} - \alpha^{(k)}\vec{\mathbf{d}}^{(k)}$$

Where $\alpha^{(k)}$ is chosen to minimize $\phi(\vec{u}^{(k)} - \alpha^{(k)}\vec{\mathbf{d}}^{(k)})$ with respect to $\alpha^{(k)}$.

13.3.1 Steepest Descent

The direction of steepest descent of a surface is $-\nabla\phi(\vec{\mathbf{u}}^{(k)})$. This turns out to be the residual:

$$-\nabla\phi(\vec{\mathbf{u}}^{(k)}) = A\vec{\mathbf{u}}^{(k)} - \vec{\mathbf{b}} = -\vec{\mathbf{r}}^{(k)}$$

To find the step size,

$$\alpha^{(k)} = \underset{[}{\operatorname{argmin}}\, \alpha]\phi(\vec{u}^{(k)} + \alpha^{(k)}\vec{\mathbf{r}}^{(k)})$$

This is done via

$$\phi(\vec{\mathbf{u}} + \alpha\vec{\mathbf{r}}) = \frac{1}{2}\vec{\mathbf{u}}^T A\vec{\mathbf{u}} - \vec{\mathbf{u}}^T b + \alpha\left(\vec{\mathbf{r}}^T A\vec{\mathbf{r}} - \vec{\mathbf{r}}^T b\right) + \frac{1}{2}\alpha^2\left(\vec{\mathbf{r}}^T A\vec{\mathbf{r}}\right)$$

Taking the derivative with respect to α

$$\alpha\left(\vec{\mathbf{r}}^T A\vec{\mathbf{r}}\right) = \vec{\mathbf{r}}^T\vec{\mathbf{b}} - \vec{\mathbf{r}}^T A\vec{\mathbf{u}} = \vec{\mathbf{r}}^T(\vec{\mathbf{b}} - A\vec{\mathbf{u}}) = \vec{\mathbf{r}}^T\vec{\mathbf{r}}$$

So $\alpha = \frac{\vec{\mathbf{r}}^T\vec{\mathbf{r}}}{\vec{\mathbf{r}}^T A\vec{\mathbf{r}}}$ thus

$$\alpha^{(k)} = \frac{(\vec{\mathbf{r}}^{(k)})^T\vec{\mathbf{r}}^{(k)}}{(\vec{\mathbf{r}}^{(k)})^T A\vec{\mathbf{r}}^{(k)}}, \qquad \vec{\mathbf{d}} = -\vec{\mathbf{r}}$$

Note

$$\vec{\mathbf{r}}^{(k)} = \vec{\mathbf{b}} - A\vec{\mathbf{u}}^{(k)} = \vec{\mathbf{b}} - A(\vec{\mathbf{u}}^{(k-1)} + \alpha^{(k-1)}\vec{\mathbf{r}}^{(k-1)}) = \vec{\mathbf{r}}^{(k-1)} - \alpha^{(k-1)}A\vec{\mathbf{r}}^{(k-1)})$$

Note: Since A is SPD, level curves of ϕ are hyper-ellipses centered at $\vec{\mathbf{u}}^*$.
Convergence: For $m = 2$, matrices with higher eccentricity $\frac{\lambda_1}{\lambda_2}$ will take more iterations.

13.3.2 Conjugate Gradient

Guaranteed to converge to the solution in m steps.
Consider $m = 2$. Start with an initial guess and arbitrary search direction $\vec{\mathbf{d}}^{(0)}$, we want $\vec{\mathbf{d}}^{(1)} = \vec{\mathbf{u}}^* - \vec{\mathbf{u}}^{(1)}$ give $\vec{\mathbf{d}}^{(0)} = \vec{\mathbf{u}}^{(1)} - \vec{\mathbf{u}}^{(0)}$. Claim $\vec{\mathbf{d}}^{(1)}, \vec{\mathbf{d}}^{(0)}$ are A-othogonal $(\vec{\mathbf{d}}^{(0)})^T A\vec{\mathbf{d}}^{(1)} = 0$.
Proof. $\vec{\mathbf{d}}^{(0)}$ is tangent to a level curve of ϕ $\vec{\mathbf{u}}^{(1)}$. $\nabla\phi(\vec{\mathbf{u}}^{(1)}) = A\vec{\mathbf{u}}^{(1)} - \vec{\mathbf{b}} = -\vec{\mathbf{r}}^{(1)}$. $\vec{\mathbf{r}}^{(1)}$ is orthogonal to $dr^{(0)}$: $(\vec{\mathbf{d}}^{(0)})^T\vec{\mathbf{r}}^{(1)} = 0$.

$$\vec{\mathbf{r}}^{(1)} = \vec{\mathbf{b}} - A\vec{\mathbf{u}}^{(1)} = A\vec{\mathbf{u}}^* - A\vec{\mathbf{u}}^{(1)} = A\vec{\mathbf{d}}^{(1)}$$

Thus $(\vec{\mathbf{d}}^{(0)})^T\vec{\mathbf{r}}^{(1)} = (\vec{\mathbf{d}}^{(0)})^T A\vec{\mathbf{d}}^{(1)} = 0$.
Idea: At each step, pick a search direction so that $d^{(k)}$ and $d^{(k-1)}$ are A-orthogonal.

$$d^{(k)} = -r^{(k)} + \beta^{(k)}d^{(k-1)}$$

Choose $\beta^{(k)}$ so that $\vec{\mathbf{d}}^{(k)}$ and $d^{(k-1)}$ are A-orthogonal.

$$\beta^{(k)} = \frac{(\vec{\mathbf{r}}^{(k)})^T A\vec{\mathbf{d}}^{(k-1)}}{(\vec{\mathbf{d}}^{(k-1)})^T Ad^{(k-1)}} = ... = \frac{(\vec{\mathbf{r}}^{(k)})^T\vec{\mathbf{r}}^{(k)}}{(\vec{\mathbf{r}}^{(k-1)})^T\vec{\mathbf{r}}^{(k-1)}}$$

Optimal step size

$$\alpha^{(k)} = \underset{[}{\operatorname{argmin}}\, \alpha]\phi(\vec{\mathbf{u}}^{(k)} + \alpha r^{(k)} - \alpha\beta d^{(k-1)} = \frac{(\vec{\mathbf{d}}^{(k)})^T\vec{\mathbf{r}}^{(k)}}{(\vec{\mathbf{d}}^{(k)})^T A\vec{\mathbf{d}}^{(k)}}$$

Note: $A \in \mathbb{R}^{n \times n}$ is SPD. Converges in m steps to exact solution.
Converses to $\|\vec{\mathbf{r}}\| \ < \epsilon$ in $O(()\sqrt{\kappa(A)})$. Poisson in 2D: $\kappa(A) = O(()h^{-2}) = O(()N^2)$. Iterations will be $O(()N)$