

Projection and its Importance in Scientific Computing

Stan Tomov

EECS Department

The University of Tennessee, Knoxville

March 22, 2017

- Contact information

office : Claxton 317

phone : (865) 974-6317

email : tomov@cs.utk.edu

Additional reference materials:

- [1] R.Barrett, M.Berry, T.F.Chan, J.Demmel, J.Donato, J. Dongarra, V. Eijkhout, R.Pozo, C.Romine, and H.Van der Vorst, *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods (2nd edition)*
http://netlib2.cs.utk.edu/linalg/html_templates/Templates.html
- [2] Yousef Saad, *Iterative methods for sparse linear systems (1st edition)*
<http://www-users.cs.umn.edu/~saad/books.html>

Topics

as related to **high-performance scientific computing** □

Projection in Scientific Computing

Sparse matrices,
parallel implementations

PDEs, Numerical
solution, Tools, etc.

Iterative Methods

Topics

on new architectures – multicore, GPUs (CUDA & OpenCL), MIC

Projection in Scientific Computing

Sparse matrices,
parallel implementations

PDEs, Numerical
solution, Tools, etc.

Iterative Methods

Outline

- Part I
 - Fundamentals
- Part II
 - Projection in Linear Algebra
- Part III
 - Projection in Functional Analysis (e.g. PDEs)
- HPC with Multicore and GPUs

Part I

Fundamentals

Projection in Scientific Computing

[an example – in solvers for PDE discretizations]

Electronic structure calculations

- Density functional theory

Many-body Schrödinger equation (exact but exponential scaling)

$$\left\{ -\sum_i \frac{1}{2} \nabla_i^2 + \sum_{i,j} \frac{1}{|r_i - r_j|} + \sum_{i,j} \frac{Z}{|r_i - R_j|} \right\} \Psi(r_1, \dots, r_N) = E \Psi(r_1, \dots, r_N)$$

- Nuclei fixed, generating external potential (system dependent, non-trivial)
- N is number of electrons



Kohn Sham Equation: The many body problem of interacting electrons is reduced to non-interacting electrons (single particle problem) with the same electron density and a different effective potential (cubic scaling).

$$\left\{ -\frac{1}{2} \nabla^2 + \int \frac{\rho(r')}{|r - r'|} dr' + \sum_i \frac{Z}{|r - R_i|} + V_{xc} \right\} \psi_i(r) = E_i \psi_i(r)$$

$$\rho(r) = \sum_i |\psi_i(r)|^2 = |\Psi(r_1, \dots, r_N)|^2$$

- V_{xc} represents effects of the Coulomb interactions between electrons
- ρ is the density (of the original many-body system)

V_{xc} is not known except special cases \Rightarrow use approximation, e.g. Local Density Approximation (LDA)

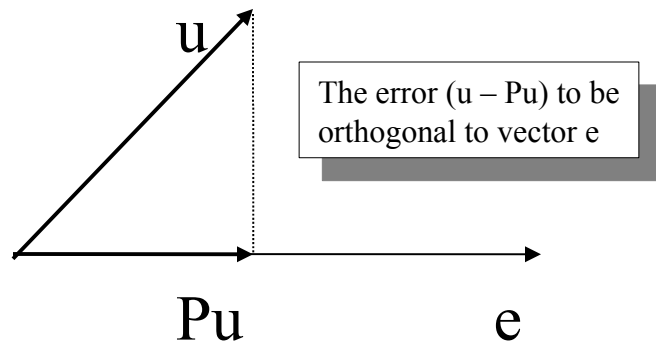
where V_{xc} depends only on ρ

- A model leading to self-consistent iteration with need for high-performance diagonalization and orthogonalization routines

What is Projection?

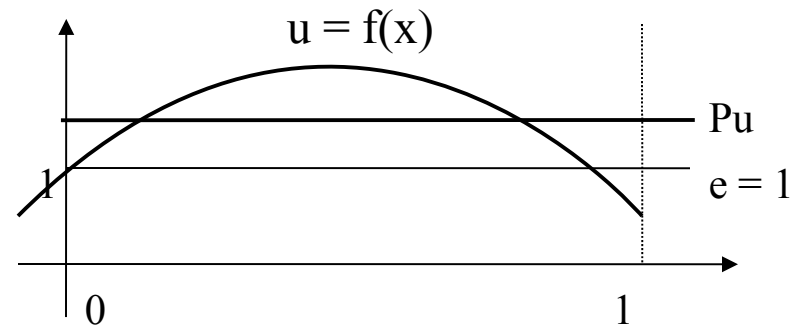
- Here are two examples

(from linear algebra)



P : orthogonal projection of vector u on e

(from functional analysis)



P : best approximation (projection) of $f(x)$ in $\text{span}\{e\} \subset C[0,1]$

Definition

- **Projection** is a linear transformation P from a linear space V to itself such that
$$P^2 = P$$

equivalently

Let V is direct sum of subspaces V_1 and V_2

$$V = V_1 \oplus V_2$$

i.e. for $\forall u \in V$ there are unique $u_1 \in V_1$ and $u_2 \in V_2$ s.t.

$$u = u_1 + u_2$$

Then $P: V \rightarrow V_1$ is defined for $\forall u \in V$ as $Pu \equiv u_1$

Importance in Scientific Computing

- To compute approximations $Pu \approx u$ where $\mathbf{dim} V_1 \ll \mathbf{dim} V$
$$V = V_1 \oplus V_2$$
- When computation directly in V is not feasible or even possible.

A few examples:

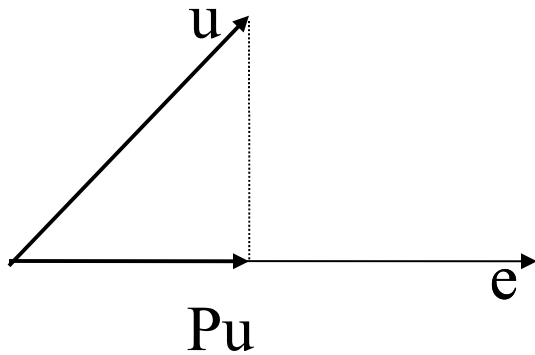
- Interpolation (via polynomial interpolation/projection)
- Image compression
- Sparse iterative linear solvers and eigensolvers
- Finite Element/Volume Method approximations
- Least-squares approximations, etc.

Projection in \mathbb{R}^2

- In \mathbb{R}^2 with Euclidean inner-product, i.e. for $\mathbf{x}, \mathbf{y} \in \mathbb{R}^2$

$$(\mathbf{x}, \mathbf{y}) = x_1 y_1 + x_2 y_2 \quad (= \mathbf{y}^T \mathbf{x} = \mathbf{x} \cdot \mathbf{y})$$

$$\text{and } \|\mathbf{x}\| = (\mathbf{x}, \mathbf{x})^{1/2}$$



P : orthogonal projection of
vector u on e

$$Pu = \frac{(\mathbf{u}, \mathbf{e})}{\|\mathbf{e}\|^2} \mathbf{e} \quad (\text{Exercise})$$

i.e. for $\|\mathbf{e}\| = 1$

$$Pu = (\mathbf{u}, \mathbf{e}) \mathbf{e}$$

Projection in $\mathbb{R}^n / \mathbb{C}^n$

- Similarly to \mathbb{R}^2

P : Orthogonal projection of u into $\text{span}\{e_1, \dots, e_m\}$, $m \leq n$.

Let e_i , $i = 1 \dots m$ is **orthonormal** basis, i.e.

$$(e_i, e_j) = 0 \quad \text{for } i \neq j \text{ and}$$

$$(e_i, e_j) = 1 \quad \text{for } i=j$$

$$P u = \underline{(u, e_1) e_1} + \dots + (u, e_m) e_m \quad (\text{Exercise})$$



Orthogonal projection of u on e_1

How to get an orthonormal basis?

- Can get one from every subspace by **Gram-Schmidt** orthogonalization:

Input : m linearly independent vectors x_1, \dots, x_m

Output : m orthonormal vectors x_1, \dots, x_m

1. $x_1 = x_1 / \|x_1\|$
2. do i = 2, m
3. $x_i = x_i - \underbrace{(x_i, x_1)}_{\text{Orthogonal projection of } x_i \text{ on } x_1} x_1 - \dots - (x_i, x_{i-1}) x_{i-1}$ (Exercise: $x_i \perp x_1, \dots, x_{i-1}$)
4. $x_i = x_i / \|x_i\|$
5. enddo

Known as **Classical Gram-Schmidt** (CGM) orthogonalization

How to get an orthonormal basis?

- What if we replace line 3 with the following (3')?

$$3. \quad x_i = x_i - (x_i, x_1) x_1 - \dots - (x_i, x_{i-1}) x_{i-1}$$

$$3'. \quad \text{do } j = 1, i-1$$

$$x_i = x_i - (x_i, x_j) x_j$$

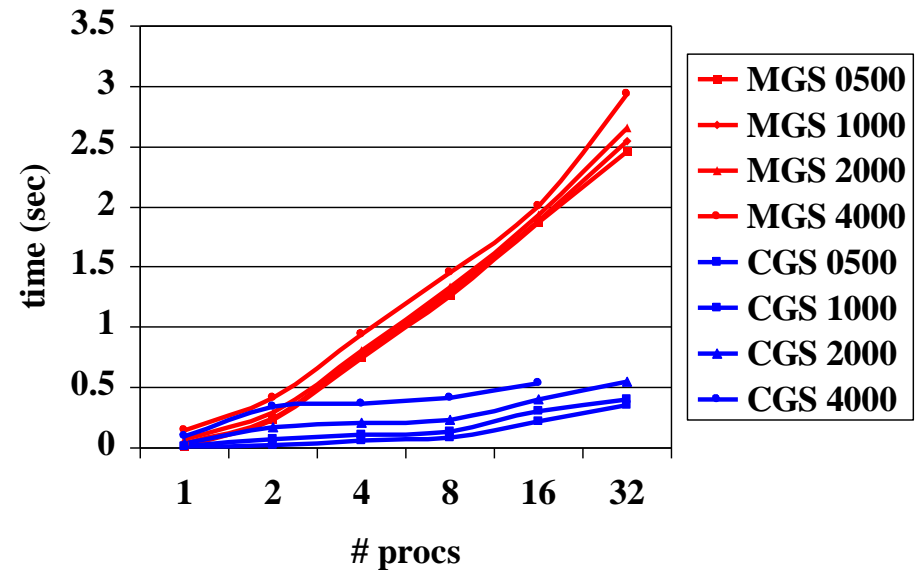
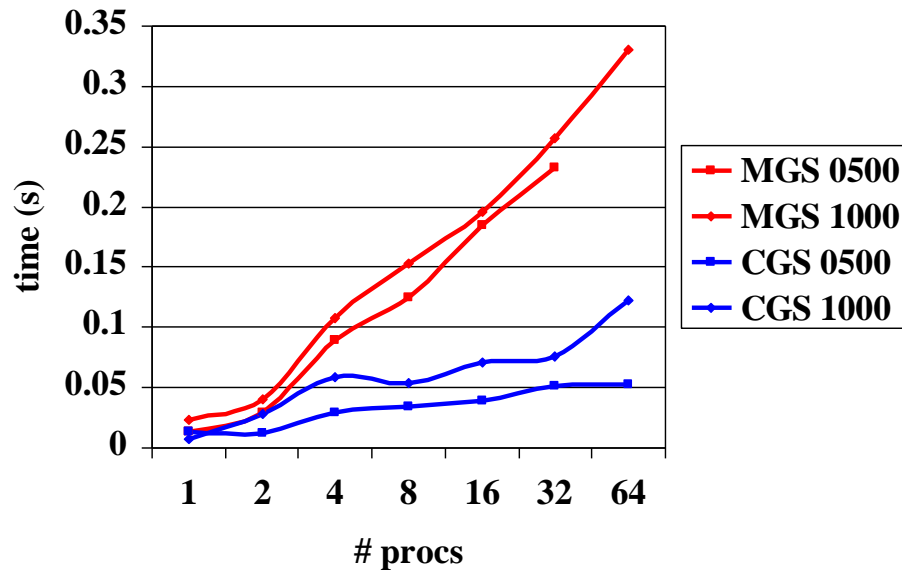
enddo

- Equivalent in exact arithmetic (**Exercise**) but not with round-off errors (next) !
- Known as **Modified Gram-Schmidt** (MGS) orthogonalization

CGS vs MGS

[Results from Julien Langou:]

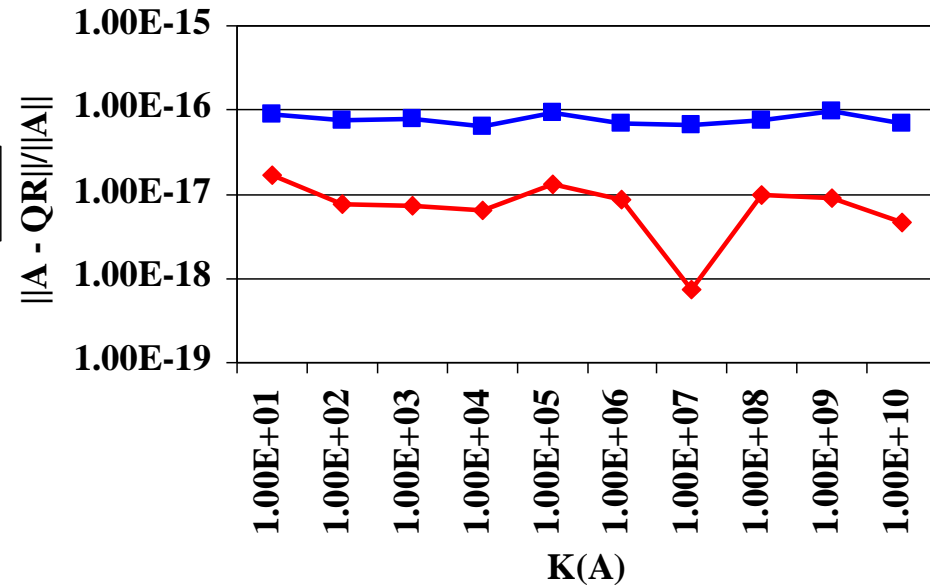
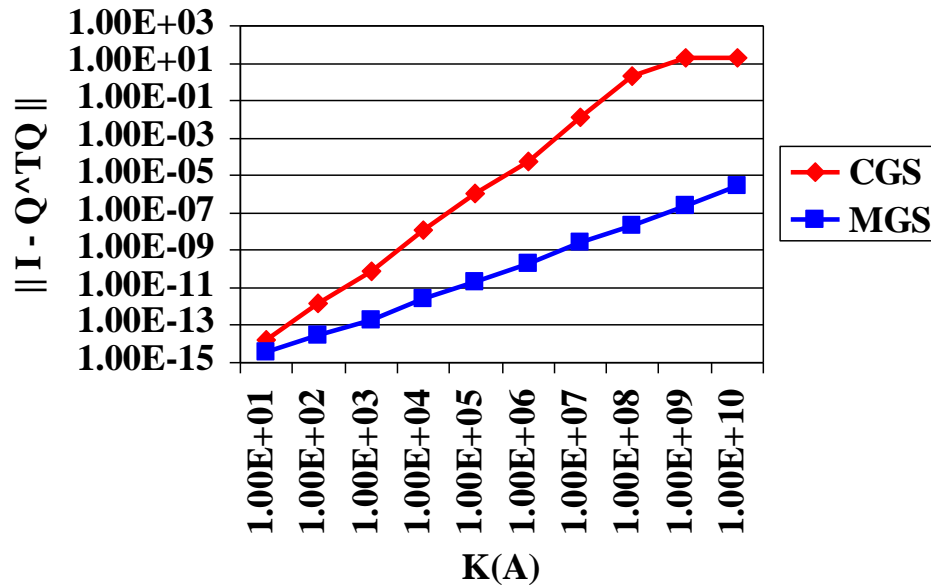
Scalability of MGS and CGS on two different clusters for matrices of various size
 $m=[500 \ 1000 \ 2000 \ 4000]$ per processor, $n = 100$



CGS vs MGS

[Results from Julien Langou:]

Accuracy of MGS vs CGS on matrices of increasing condition number



QR factorization

- Let $\mathbf{A} = [\mathbf{x}_1, \dots, \mathbf{x}_m]$ be the input for CGS/MGS and
 $\mathbf{Q} = [\mathbf{q}_1, \dots, \mathbf{q}_m]$ the output;

\mathbf{R} : an upper $m \times m$ triangular matrix defined from the CGR/MGS.

Then

$$\mathbf{A} = \mathbf{Q} \mathbf{R}$$

Other QR factorizations

- **What about the following?**

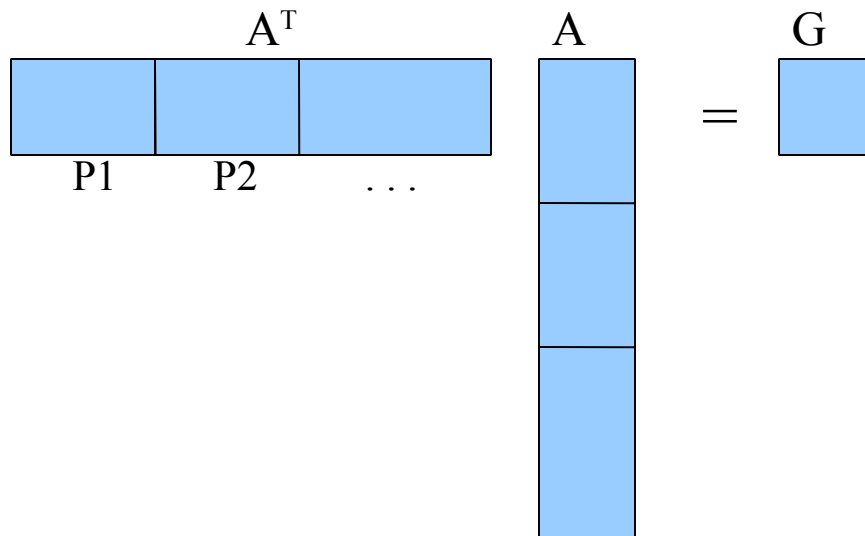
[known as Cholesky QR]

1. $G = A^T A$
2. $G = L L^T$ (Cholesky factorization)
3. $Q = A (L^T)^{-1}$

- Does Q have orthonormal columns (i.e. $Q^T Q = I$), i.e. $A = Q L^T$ to be a QR factorization (**Exercise**)
- When is this feasible and how compares to CGS and MGS?

Other QR factorizations

- Feasible when $n \gg m$
- Allows efficient parallel implementation:
blocking both computation and communication



**Investigate numerical accuracy
and scalability
(compare to CGS and MGS)
Exercise**

How is done in LAPACK?

- Using **Householder reflectors**

$$H = I - 2 w w^T$$

- $w = ?$ so that

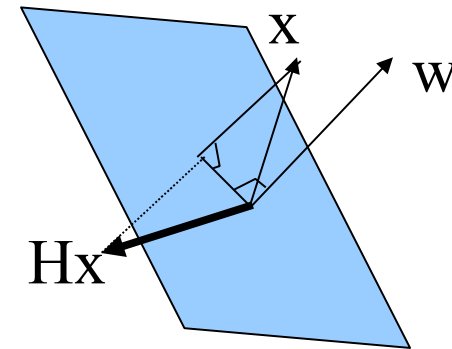
$$H x_1 = \alpha e_1$$

$\Rightarrow w = \dots$ (compute or look at the reference books)

- Allows us to construct

$$X_k \equiv \underbrace{H_{k-1} \dots H_1}_Q X = \begin{pmatrix} x_{11} & x_{12} & x_{13} & \dots & \dots & \dots & x_{1m} \\ & x_{22} & x_{23} & \dots & \dots & \dots & x_{2m} \\ & & x_{33} & \dots & \dots & \dots & x_{3m} \\ & & & \ddots & \dots & \dots & \vdots \\ & & & & x_{kk} & \dots & \vdots \\ & & & & x_{k+1,k} & \dots & x_{k+1,m} \\ & & & & \vdots & \vdots & \vdots \\ & & & & x_{n,k} & \dots & x_{n,m} \end{pmatrix}$$

(Exercise)



LAPACK implementation : “**delayed update**” of the trailing matrix +
 “**accumulate transformation**” to apply it as **BLAS 3**

Part II

Projection in Linear Algebra

Projection into general basis

- How to define projection without orthogonalization of a basis?
 - Sometimes is not feasible to orthogonalize
 - Often the case in functional analysis
(e.g. Finite Element Method, Finite Volume Method, etc.)
where the basis is “linearly independent” functions (more later, and Lecture 2)

- We saw if $X = [x_1, \dots, x_m]$ is an orthonormal basis

$$(*) \quad \mathbf{P} \mathbf{u} = (\mathbf{u}, \mathbf{x}_1) \mathbf{x}_1 + \dots + (\mathbf{u}, \mathbf{x}_m) \mathbf{x}_m$$

- How does (*) change if X are just linearly independent ?

$$\mathbf{P} \mathbf{u} = \quad ?$$

Projection into a general basis

- The problem:

Find the coefficients $C = (c_1 \dots c_m)^T$ in

$$P u = c_1 x_1 + c_2 x_2 + \dots + c_m x_m = X C$$

so that

$$u - P u \perp \text{span}\{x_1, \dots, x_m\}$$

or \Leftrightarrow so that the **error** e in

$$(1) \quad u = P u + e$$

is $\perp \text{span}\{x_1, \dots, x_m\}$

Projection into a general basis

- (1) $\mathbf{u} = \mathbf{P} \mathbf{u} + \mathbf{e} = \mathbf{c}_1 \mathbf{x}_1 + \mathbf{c}_2 \mathbf{x}_2 + \dots + \mathbf{c}_m \mathbf{x}_m + \mathbf{e}$
- Multiply (1) on both sides by “*test*” vector/function x_j (terminology from functional analysis) for $j = 1, \dots, m$

$$(\mathbf{u}, \mathbf{x}_j) = \mathbf{c}_1 (\mathbf{x}_1, \mathbf{x}_j) + \mathbf{c}_2 (\mathbf{x}_2, \mathbf{x}_j) + \dots + \mathbf{c}_m (\mathbf{x}_m, \mathbf{x}_j) + \cancel{(\mathbf{e}, \mathbf{x}_j)}$$

- i.e., m equations for m unknowns
- In matrix notations $\Leftrightarrow (\mathbf{X}^T \mathbf{X}) \mathbf{C} = \mathbf{X}^T \mathbf{u}$ (Exercise)
 $\Leftrightarrow \mathbf{X}^T \mathbf{P} \mathbf{u} = \mathbf{X}^T \mathbf{u}$
- $\mathbf{X}^T \mathbf{X}$ is the so called Gram matrix (nonsingular; **why?**) \Rightarrow there exists a unique solution \mathbf{C}

Normal equations

- System

$$(\mathbf{X}^T \mathbf{X}) \mathbf{C} = \mathbf{X}^T \mathbf{u}$$

is known also as **Normal Equations**

- The *Method of Normal Equations*:
Finding the projection (approximation) $\mathbf{P}\mathbf{u} \equiv \mathbf{X}\mathbf{C}$ (approximation) of \mathbf{u} in \mathbf{X} by solving the Normal Equations system

Least Squares (LS)

- Equivalently, system

$$(\mathbf{X}^T \mathbf{X}) \mathbf{c} = \mathbf{X}^T \mathbf{u}$$

gives also the solution of the LS problem

$$\min \|\mathbf{X} \mathbf{c} - \mathbf{u}\|$$

since $\|\mathbf{v}_1 - \mathbf{u}\|^2 \stackrel{\mathbf{c} \in \mathbb{R}^m}{=} \|(\mathbf{v}_1 - \mathbf{P}\mathbf{u}) - \mathbf{e}\|^2 = \|\mathbf{v}_1 - \mathbf{P}\mathbf{u}\|^2 + \|\mathbf{e}\|^2$
 $\geq \|\mathbf{e}\|^2 = \|\mathbf{P}\mathbf{u} - \mathbf{u}\|^2$ for $\forall \mathbf{v}_1 \in V_1$

LS

- Note that the usual notations for LS is: For $A \in \mathbb{R}^{n \times m}$, $b \in \mathbb{R}^n$ find

$$\min_{x \in \mathbb{R}^m} \| A x - b \|$$

- Solving LS with QR factorization

$$\text{Let } A = Q R, \quad Q^T A = R = \begin{pmatrix} R_1 \\ 0 \end{pmatrix}, \quad Q^T b = \begin{pmatrix} c \\ d \end{pmatrix} \quad \begin{matrix} m \\ n - m \end{matrix}$$

Then

$$\| Ax - b \|^2 = \| Q^T Ax - Q^T b \|^2 = \| R_1 x - c \|^2 + \| d \|^2$$

i.e. we get minimum if x is such that

$$R_1 x = c$$

Projection and iterative solvers

- The problem : Solve
$$Ax = b \quad \text{in } \mathbb{R}^n$$
- Iterative solution: at iteration i extract an approximate x_i from just a subspace $V = \text{span}[v_1, \dots, v_m]$ of \mathbb{R}^n
- How? As on slide 22, impose constraints:
$$b - Ax \perp \text{subspace } W = \text{span}[w_1, \dots, w_m] \text{ of } \mathbb{R}^n, \text{ i.e.}$$
$$(*) \quad (Ax, w_i) = (b, w_i) \quad \text{for } \forall w_i \in W = \text{span}[w_1, \dots, w_m]$$
- Conditions (*) known also as **Petrov-Galerkin conditions**
- Projection is **orthogonal**: V and W are the same (Galerkin conditions) or **oblique** : V and W are different

Matrix representation

- Let $V = [v_1, \dots, v_m]$, $W = [w_1, \dots, w_m]$
Find $y \in \mathbb{R}^m$ s.t. $x = x_0 + V y$ solves $Ax = b$, i.e.
 $A V y = b - Ax_0 = r_0$

subject to the orthogonality constraints:

$$W^T A V y = W^T r_0$$

- The choice for V and W is crucial and determines various methods (more in Lectures 13 and 14)

A General Projection Algorithm

- Prototype from Y.Saad's book
 1. *Until convergence, Do:*
 2. *Select a pair of subspaces \mathcal{K} and \mathcal{L}*
 3. *Choose bases $V = [v_1, \dots, v_m]$ and $W = [w_1, \dots, w_m]$ for \mathcal{K} and \mathcal{L}*
 4. $r := b - Ax$
 5. $y := (W^T AV)^{-1} W^T r$
 6. $x := x + Vy$
 7. *EndDo*

Projection and Eigen-Solvers

- The problem : Solve
$$Ax = \lambda x \quad \text{in } \mathbb{R}^n$$
- As in linear solvers: at iteration i extract an approximate x_i from a subspace $V = \text{span}[v_1, \dots, v_m]$ of \mathbb{R}^n
- How? As on slides 22 and 26, impose constraints:
$$\lambda x - Ax \perp \text{subspace } W = \text{span}[w_1, \dots, w_m] \text{ of } \mathbb{R}^n, \text{ i.e.}$$
$$(*) \quad (Ax, w_i) = (\lambda x, w_i) \quad \text{for } \forall w_i \in W = \text{span}[w_1, \dots, w_m]$$
- This procedure is known as **Rayleigh-Ritz**
- Again projection can be *orthogonal* or *oblique*

Matrix representation

- Let $V = [v_1, \dots, v_m]$, $W = [w_1, \dots, w_m]$
Find $y \in \mathbb{R}^m$ s.t. $\mathbf{x} = V \mathbf{y}$ solves $A \mathbf{x} = \lambda \mathbf{x}$, i.e.
 $A V \mathbf{y} = \lambda V \mathbf{y}$

subject to the orthogonality constraints:

$$W^T A V \mathbf{y} = \lambda W^T V \mathbf{y}$$

- The choice for V and W is crucial and determines various methods (more in Lectures 4 and 5)

Part III

Projection in PDEs

Projection in Functional Spaces

- The discussion so far can be applied to any functional inner-product space (examples to follow)
- An important space is $\mathbf{C}[\mathbf{a}, \mathbf{b}]$, the space of continuous functions on $[\mathbf{a}, \mathbf{b}]$, with inner-product

$$(f, g) = \int_a^b f(x) g(x) dx$$

and induced norm

$$\| f \| = (f, f)^{1/2}$$

Projection in Functional Spaces

- Projection $P: V \rightarrow V_1$ where $V = V_1 \oplus V_2$
- In functional analysis and scientific computing V_1 is usually taken as
 - Piecewise polynomials
 - In PDE approximation (FEM/FVM), Numerical integration, etc.
 - Trigonometric functions
 - $\{ \sin(n x), \cos(n x) \}_{n=0, \dots}$, $x \in [0, 2\pi]$

Orthogonal relative to $\int_0^{2\pi}$

$$(f, g) = \int_0^{2\pi} f(x) g(x) dx \quad (\text{Exercise})$$

Normal equations / LS

- **Exercise:**

$$f(x) = \sin(x)$$

Find the projection in $V_1 = \text{span}\{x, x^3, x^5\}$ on interval $[-1, 1]$ using inner-product

$$(f, g) = \int_{-1}^1 f(x) g(x) \, dx$$

and norm $\| f \| = (f, f)^{1/2}$

Normal equations / LS

- Leads to Gram matrix that is very ill-conditioned (called Hilbert matrix: Gram matrix for polynomials $1, x, x^2, x^3, \dots$)
- For numerical stability is better to orthogonalize the polynomials
- There are numerous examples of orthonormal polynomial sets
 - * Legendre, Chebyshev, Hermite, etc.
 - * Check the literature for more if interested

Integration via Polynomial Interpolation

- Take

$$\int f(x) \, dx \approx \int p(x) \, dx$$

where p is a polynomial approximation to f

- Taking p a polynomial interpolating f at $n+1$ fixed nodes x_i leads to quadrature formulas

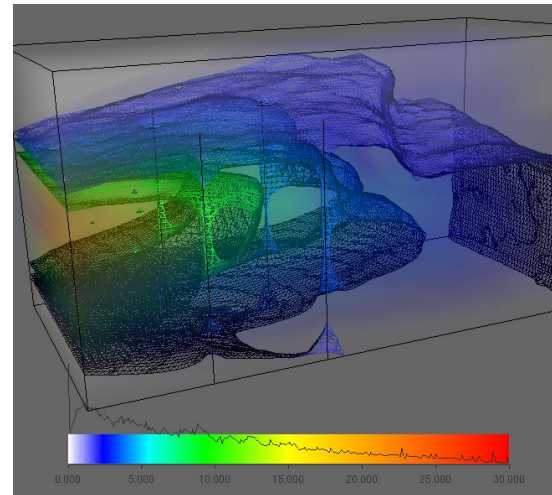
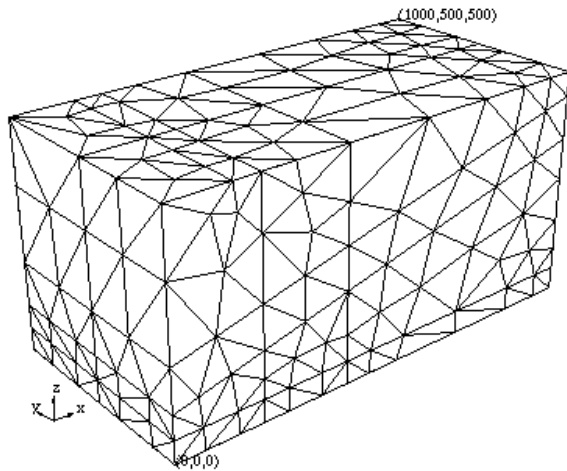
$$\int f(x) \, dx \approx A_0 f(x_0) + \dots + A_n f(x_n)$$

that are exact exact for polynomials of degree $\leq n$

- Smart choice of the nodes x_i (Gaussian quadrature) leads to formulas that are exact for polynomials of degree $\leq 2n+1$

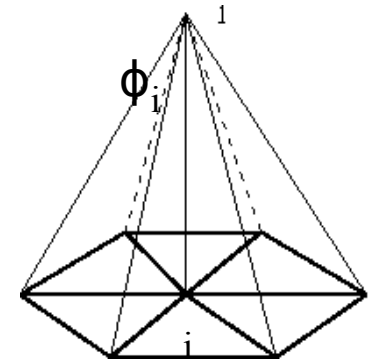
Galerkin Projection

- Numerical PDE discretizations have a common concept:
 - Represent computational domain with mesh
 - Approximate functions and operators over the mesh



Galerkin Projection

- Finite dimensional spaces (e.g. V_1) can be piecewise polynomials defined over the mesh, e.g.
- Numerical solution of PDE (e.g. FEM)



- Boundary value problem: $Au = f$, subject to boundary conditions
- Get a “weak” formulation: $(Au, \phi) = (f, \phi)$ - multiply by test function ϕ and integrate over the domain

$$a(u, \phi) = \langle f, \phi \rangle \text{ for } \forall \phi \in S$$

- Galerkin (FEM) problem: Find $u_h \in S_h \subset S$ s.t.

$$a(u_h, \phi_h) = \langle f, \phi_h \rangle \text{ for } \forall \phi_h \in S_h$$

Learning Goals

- To refresh some linear algebra essentials that are of fundamental importance for scientific computing
- The idea and application of **Petrov-Galerkin conditions** as a way of defining computationally feasible formulations (approximations)
- Some generic examples demonstrating the ideas in
 - Linear algebra
 - Functional analysis
(to get more specific in the following lectures)