

# MA5233 Computational Mathematics

## Lecture 5: Poisson Equation

Simon Etter



2019/2020

# Poisson Equation

## Poisson equation

- ▶ Linear systems often arise as discretisations of partial differential equations (PDEs).
- ▶ These linear systems have a special structure which allows for much faster algorithms.
- ▶ Poisson equation is the most famous of all PDEs.

## Outline

- ▶ Introduction to the Poisson equation
- ▶ Solving the Poisson equation using finite differences
- ▶ Convergence of the finite difference discretisation

# Poisson Equation

## Poisson equation

Given  $\Omega \subset \mathbb{R}^n$  and  $f : \Omega \rightarrow \mathbb{R}$ , find  $u : \Omega \rightarrow \mathbb{R}$  such that

$$\begin{cases} -\Delta u(x) = f(x) & \forall x \in \Omega, \\ u(x) = 0 & \forall x \in \partial\Omega. \end{cases}$$

$\Delta := \frac{\partial^2}{\partial x_1^2} + \dots + \frac{\partial^2}{\partial x_n^2}$  is called the Laplacian operator.

## Physical interpretation

	$f$	$u$
diffusion	particle source / sink	concentration
heat	heat source / sink	temperature
electrostatics	charge distribution	potential

# Poisson Equation

## Derivation of the Poisson equation

- Fick's law of diffusion: net flux is given by

$$\vec{J} = -D \nabla u = -D \begin{pmatrix} \frac{\partial u}{\partial x_1} \\ \vdots \\ \frac{\partial u}{\partial x_n} \end{pmatrix}.$$

- Conservation of mass:

$$\frac{\partial}{\partial t} \int_{\Omega'} u \, dx = - \int_{\partial\Omega'} \vec{n} \cdot \vec{J} \, dx + \int_{\Omega'} f \, dx.$$

- Divergence / Gauss's theorem:

$$\int_{\partial\Omega'} \vec{n} \cdot \vec{J} \, dx = \int_{\Omega'} \nabla \cdot \vec{J} \, dx = \int_{\Omega'} \left( \frac{\partial J_1}{\partial x_1} + \dots + \frac{\partial J_n}{\partial x_n} \right) dx$$

# Poisson Equation

## Derivation of the Poisson equation

- ▶ Combining conservation of mass with divergence theorem yields

$$\int_{\Omega'} \left( \frac{\partial u}{\partial t} + \nabla \cdot \vec{J} - f \right) dx = 0 \quad \forall \Omega' \subset \Omega.$$

Hence

$$\frac{\partial u}{\partial t} + \nabla \cdot \vec{J} - f = 0.$$

- ▶ Inserting steady-state condition  $\frac{\partial u}{\partial t} = 0$  and Fick's law yields

$$-\nabla \cdot (D \nabla u) = f.$$

In electrostatics,  $D$  represents the electrical permittivity of a given material.

# Poisson Equation

## Terminology

Equations:

- ▶ Poisson equation:  $-\Delta u = f$  (elliptic)
- ▶ Laplace equation:  $-\Delta u = 0$  (elliptic)
- ▶ Heat equation:  $\frac{\partial u}{\partial t} = \Delta u + f$  (parabolic)

Boundary conditions:

- ▶ homogeneous Dirichlet:  $u(x) = 0$
- ▶ inhomogeneous Dirichlet:  $u(x) = g(x)$
- ▶ Neumann:  $\vec{n} \cdot \nabla u = g(x)$

# Poisson Equation

## Physical interpretation of boundary conditions

Homogeneous Dirichlet,  $u(x) = 0$ :

- ▶ Diffusion: particles reaching  $\partial\Omega$  get trapped.
- ▶ Heat: constant temperature on boundary.
- ▶ Electrostatics: constant potential on boundary.

Neumann,  $\vec{n} \cdot \nabla u(x) = g(x)$ :

- ▶ Diffusion & heat: prescribed flux across boundary.
- ▶ Electrostatics: prescribed charges on boundary.

# Poisson Equation

## Handling inhomogeneous Dirichlet boundary conditions

$$\begin{cases} -\Delta u = f & \text{on } \Omega \\ u = g & \text{on } \partial\Omega \end{cases}$$

is equivalent to  $u = u_0 + g$  where

$$\begin{cases} -\Delta u_0 = f + \Delta g & \text{on } \Omega \\ u_0 = 0 & \text{on } \partial\Omega \end{cases}$$



# Poisson Equation

## Example: heat flow through wall

- ▶ Consider wall of thickness  $L$  with diffusion constant  $D$ .
- ▶ Assume there is a temperature difference  $\Delta T$  across the wall.
- ▶ Let  $u(x)$  denote the temperature as function of distance  $x$  to the left end of the wall.

Temperature distribution  $u(x)$  satisfies the 1D Poisson equation

$$\begin{cases} -\frac{\partial}{\partial x} D \frac{\partial}{\partial x} u = 0 & \text{on } \Omega = [0, L], \\ u(0) = 0, & u(L) = \Delta T. \end{cases}$$

Solution  $u(x)$  and heat flux  $J$  are given by, respectively,

$$u(x) = \frac{\Delta T}{L} x \quad \text{and} \quad J = -\frac{D \Delta T}{L}.$$

# Poisson Equation

## Discretising the Poisson equation

Assuming  $\Omega = [0, 1]$ ,  $u(0) = u(1) = 0$  and  $D = 1$  for now.

Functions:

- ▶ Introduce mesh  $\Omega_n := \{x_k := \frac{k}{n+1} \mid k = 0, \dots, n+1\}$ .
- ▶ Replace function  $f(x)$  with vector of point values  $f_k := f(x_k)$ .

Derivatives:

$$(\nabla_n u)_{k+\frac{1}{2}} := \frac{f_{k+1} - f_k}{1/(n+1)} \approx \nabla u(x_{k+\frac{1}{2}})$$

$$(\Delta_n u)_k := \frac{(\nabla_n u)_{k+\frac{1}{2}} - (\nabla_n u)_{k-\frac{1}{2}}}{1/(n+1)} \approx \Delta u(x_k)$$

Second derivative simplifies to

$$(\Delta_n u)_k = (n+1)^2 (u_{k+1} - 2u_k + u_{k-1}).$$

# Poisson Equation

## Discretising the Poisson equation

Replacing  $\Delta$  with  $\Delta_n$  in  $-\Delta u = f$  yields

$$(n+1)^2 \begin{pmatrix} 2 & -1 & & & \\ -1 & \ddots & \ddots & & \\ & \ddots & \ddots & -1 & \\ & & -1 & 2 \end{pmatrix} \begin{pmatrix} u(\frac{1}{n+1}) \\ \vdots \\ u(\frac{n}{n+1}) \end{pmatrix} = \begin{pmatrix} f(\frac{1}{n+1}) \\ \vdots \\ f(\frac{n}{n+1}) \end{pmatrix}.$$

Note that boundary points  $u(\frac{0}{n+1})$  and  $u(\frac{n+1}{n+1})$  do not appear in linear system since we already know these values.

See `5_poisson_equation.jl` for how to do this in Julia.

## Remarks

- Replacing

$$\frac{\partial u}{\partial x} \longrightarrow \frac{u(x + \frac{\Delta x}{2}) - u(x - \frac{\Delta x}{2})}{\Delta x}$$

is known as *finite difference discretisation*.

- How do we know that this is a good scheme?

# Poisson Equation

## Assessing the quality of a discretisation

Recall: A good factorisation algorithm was one which was

- ▶ accurate (relative errors are  $\mathcal{O}(\varepsilon_{\text{mach}})$ ), and
- ▶ fast (as little floating-point operations as possible).

This assessment no longer works for discretisations:

Increasing mesh size  $n$  increases FLOP count but reduces error.

New performance metric: error  $e_n$  as a function of mesh size  $n$ .

- ▶ Good algorithm:  $e_n = \mathcal{O}(2^{-n})$
- ▶ Bad algorithm:  $e_n = \mathcal{O}(\log_2(n)^{-1})$

Hence, we want a theorem of the form  $\|u - u_n\| = \mathcal{O}(g(n))$ .

# Poisson Equation

## Notation 1

We use the same symbol  $f$  to denote both a function  $f : [0, 1] \rightarrow \mathbb{R}$  and vector of point-values  $f = (f(\frac{1}{n+1}) \dots f(\frac{n}{n+1}))^T$ . Context will clarify the intended meaning.

## Notation 2

$$\|f\|_{2,n} := \frac{1}{\sqrt{n+1}} \sqrt{\sum_{k=1}^n f(\frac{k}{n+1})^2} = \frac{\|f\|_2}{\sqrt{n+1}}.$$

- ▶ For now: extra factor  $\mathcal{O}(\frac{1}{\sqrt{n}})$  is needed to ensure that the magnitude of  $\|f\|_{2,n}$  is independent of  $n$ .
- ▶ We will see later that  $\|f\|_{2,n}$  corresponds to a trapezoidal rule discretisation of  $\sqrt{\int_0^1 f(x)^2 dx}$ , assuming  $f(0) = f(1) = 0$ .

# Poisson Equation

## Theorem

Assume function  $u$  and point-values  $u_n$  satisfy, respectively,

$$-\Delta u = f \quad \text{and} \quad -\Delta_n u_n = f.$$

Then,

$$\|u - u_n\|_{2,n} \leq \|\Delta_n^{-1}\|_{2,n} \|\Delta_n u + f\|_{2,n}.$$

*Proof.* 
$$u - u_n = \Delta_n^{-1} (\Delta_n u - \Delta_n u_n) = \Delta_n^{-1} (\Delta_n u + f).$$

## Remarks

- ▶  $\|\Delta_n^{-1}\|_{2,n}$ : conditioning of discretised system.  
Confusingly, this is commonly referred to as *stability*.
- ▶  $\|\Delta_n u + f\|_{2,n}$ : consistency of discretised system.  
Measures how well exact solution  $u$  solves the discrete problem.

## Statement to remember

Consistency and stability imply convergence!

# Poisson Equation

## Consistency of finite difference discretisation

Assume  $u \in C^4([0, 1])$ . Inserting the Taylor expansion

$$\begin{aligned}u\left(\frac{k \pm 1}{n+1}\right) &= u\left(\frac{k}{n+1}\right) \pm \frac{1}{1!} u'\left(\frac{k}{n+1}\right) \frac{1}{n+1} + \frac{1}{2!} u''\left(\frac{k}{n+1}\right) \frac{1}{(n+1)^2} \\&\quad \pm \frac{1}{3!} u'''\left(\frac{k}{n+1}\right) \frac{1}{(n+1)^3} + \frac{1}{4!} u''''\left(\frac{k}{n+1}\right) \frac{1}{(n+1)^4} + o(n^{-4})\end{aligned}$$

yields

$$\begin{aligned}(\Delta_n u)_k &= (n+1)^2 \left( u\left(\frac{k+1}{n+1}\right) - 2u\left(\frac{k}{n+1}\right) + u\left(\frac{k-1}{n+1}\right) \right) \\&= u''\left(\frac{k}{n+1}\right) + \frac{2}{4!} u''''\left(\frac{k}{n+1}\right) \frac{1}{(n+1)^2} + o(n^{-2}) \\&= -f\left(\frac{k}{n+1}\right) + \frac{2}{4!} u''''\left(\frac{k}{n+1}\right) \frac{1}{(n+1)^2} + o(n^{-2}).\end{aligned}$$

Hence, consistency error is  $\|\Delta_n u + f\|_{2,n} \leq \frac{2}{4!} \|u''''\|_{[0,1]} \frac{1}{(n+1)^2} + o(n^{-2})$ .

# Poisson Equation

## Stability of finite difference discretisation

- ▶  $\Delta_n$  is a symmetric matrix.
- ▶ For such matrices, it holds  $\|\Delta_n^{-1}\|_{2,n} = \|\Delta_n^{-1}\|_2 = |\lambda_{\min}|^{-1}$ , where  $\lambda_{\min}$  is the eigenvalue of  $\Delta_n$  of smallest magnitude.
- ▶ Hence, we are looking for  $u$  such that  $\Delta_n u = \lambda u$ .

Outline for the following slides:

- ▶ Consider continuous case  $\Delta u(x) = \lambda u(x)$  for inspiration.
- ▶ Guess discrete eigenvectors based on continuous eigenfunctions.
- ▶ Verify  $\|\Delta_n^{-1}\|_{2,n} \geq C$  for some  $C$  independent of  $n$ .



# Poisson Equation

## Eigenfunctions of continuous Laplacian $\Delta$

Observations:

- ▶  $\frac{\partial^2 u}{\partial x^2}(x) = \lambda u(x)$  is satisfied by

$$\lambda = -\pi^2 \ell^2 \quad \text{and} \quad \begin{cases} u(x) = \sin(\pi \ell x), \\ u(x) = \cos(\pi \ell x). \end{cases}$$

- ▶ Only  $u(x) = \sin(\pi \ell x)$  with  $\ell \in \{1, 2, \dots\}$  satisfies  $u(0) = u(1) = 0$ .

Hence, eigenpairs of  $\Delta$  are given by

$$\lambda_\ell := -\pi^2 \ell^2 \quad \text{and} \quad u_\ell(x) := \sin(\pi \ell x).$$

We observe  $\lambda_{\min} = -\pi^2 \neq 0$ .

# Poisson Equation

## Eigenfunctions of discrete Laplacian $\Delta_n$

Educated guess: eigenvectors of  $\Delta_n$  are given by  $(u_\ell)_k := \sin\left(\pi \frac{\ell k}{n+1}\right)$ .

To verify this, it is convenient to split

$$(u_\ell)_k = \frac{1}{2i} \left( \exp\left(\pi i \frac{\ell k}{n+1}\right) - \exp\left(-\pi i \frac{\ell k}{n+1}\right) \right) =: \frac{1}{2i} \left( (e_\ell)_k - (e_{-\ell})_k \right)$$

We compute, for  $k \in \{2, \dots, n-1\}$ ,

$$\begin{aligned} (\Delta_n e_\ell)_k &= (n+1)^2 \left( (e_\ell)_{k+1} - 2(e_\ell)_k + (e_\ell)_{k-1} \right) \\ &= (n+1)^2 \left( \exp\left(\pi i \frac{\ell(k+1)}{n+1}\right) - 2 \exp\left(\pi i \frac{\ell k}{n+1}\right) + \exp\left(\pi i \frac{\ell(k-1)}{n+1}\right) \right) \\ &= (n+1)^2 \left( \exp\left(\pi i \frac{\ell}{n+1}\right) + \exp\left(-\pi i \frac{\ell}{n+1}\right) - 2 \right) \exp\left(\pi i \frac{\ell k}{n+1}\right) \\ &= \underbrace{(n+1)^2 \left( 2 \cos\left(\pi \frac{\ell}{n+1}\right) - 2 \right)}_{\lambda_\ell} (e_\ell)_k. \end{aligned}$$

By linearity, it holds  $(\Delta_n u_\ell)_k = \lambda_\ell (u_\ell)_k$  for  $k \in \{2, \dots, n-1\}$ .

# Poisson Equation

## Eigenfunctions of discrete Laplacian $\Delta_n$

For  $k = 1$ , we have

$$(\Delta_n e_\ell)_1 = \lambda_\ell (e_\ell)_k - \exp\left(\pi i \frac{\ell 0}{n+1}\right) = \lambda_\ell (e_\ell)_k - 1.$$

Hence  $(\Delta_n u_\ell)_1 = \lambda_\ell (u_\ell)_1$ , and similarly one can show  $(\Delta_n u_\ell)_n = \lambda_\ell (u_\ell)_n$

Conclusion: eigenvalues and eigenvectors of  $\Delta_n$  are given by, respectively,

$$\lambda_\ell = (n+1)^2 \left( 2 \cos \left( \pi \frac{\ell}{n+1} \right) - 2 \right) \quad \text{and} \quad (u_\ell)_k = \sin \left( \pi \frac{\ell k}{n+1} \right)$$

for  $\ell \in \{1, \dots, n\}$ .

Minimal eigenvalue is obtained for  $\ell = 1$ :

$$\lambda_{\min} = (n+1)^2 \left( 2 \cos \left( \frac{\pi}{n+1} \right) - 2 \right) = -\pi^2 + \mathcal{O}(n^{-2}).$$

# Poisson Equation

## Convergence of finite difference discretisation

Combining the consistency and stability estimates yields

$$\|u - u_n\|_{n,2} \leq \underbrace{\pi^2}_{\|\Delta_n\|_{2,n}} \underbrace{\frac{2}{4!} \|u''''\|_{[0,1]} \frac{1}{(n+1)^2}}_{\|\Delta_n u + f\|_{2,n}} + o(n^{-2}).$$

## Applications of convergence estimate

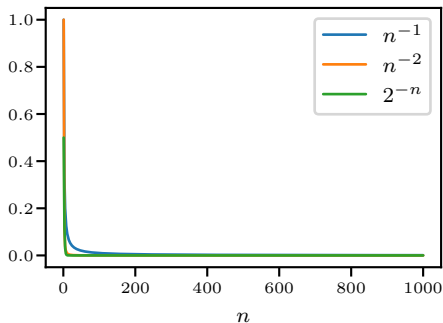
- ▶ Estimation of work required to reach sufficient accuracy.
- ▶ Code debugging!  
Try replacing the  $(n+1)^2$  scaling of  $\Delta_n$  with  $n^2$  and see what happens to the convergence rate.

# Poisson Equation

## Correct axes for convergence plots

Bad choice: linear scale for both  $x$  and  $y$  axis.

- ▶ Different decay behaviours all look the same.
- ▶ You cannot see errors  $\lesssim 10^{-2}$ .

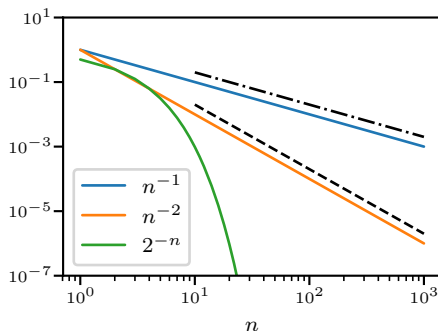


# Poisson Equation

## Correct axes for convergence plots

Good choice for algebraic decay: logarithmic scale for both  $x$  and  $y$  axis.

- ▶  $n^\alpha$  decay leads to straight line.
- ▶ Add reference lines (black lines below) so order of decay  $\alpha$  can be easily inferred.



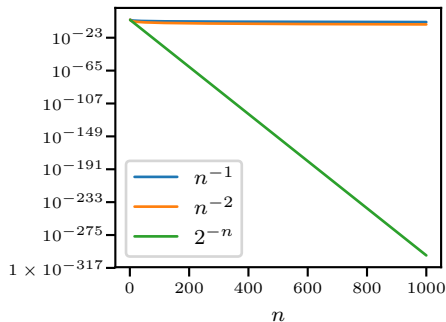
# Poisson Equation

## Correct axes for convergence plots

Good choice for exponential decay:

linear scale for  $x$  axis, logarithmic scale for  $y$  axis.

- ▶  $a^{-n}$  decay leads to straight line.
- ▶ If there is an estimate for  $a$  from theory, add reference line for comparison.



# Poisson Equation

## References and further reading

- ▶ Consistency and stability theorem:

<http://www-users.math.umn.edu/~arnold/papers/stability.pdf>