Lecture 4:

Grid-point discretisation

Discretisation

When using digital computers variables, functions, operators etc. must be represented DISCRETELY.

→ an extra source of approximation in our models

Must therefore consider extra issues:

- Discretisation error
- Stability of numerical method

We will consider 2 broad classes of discretisation;

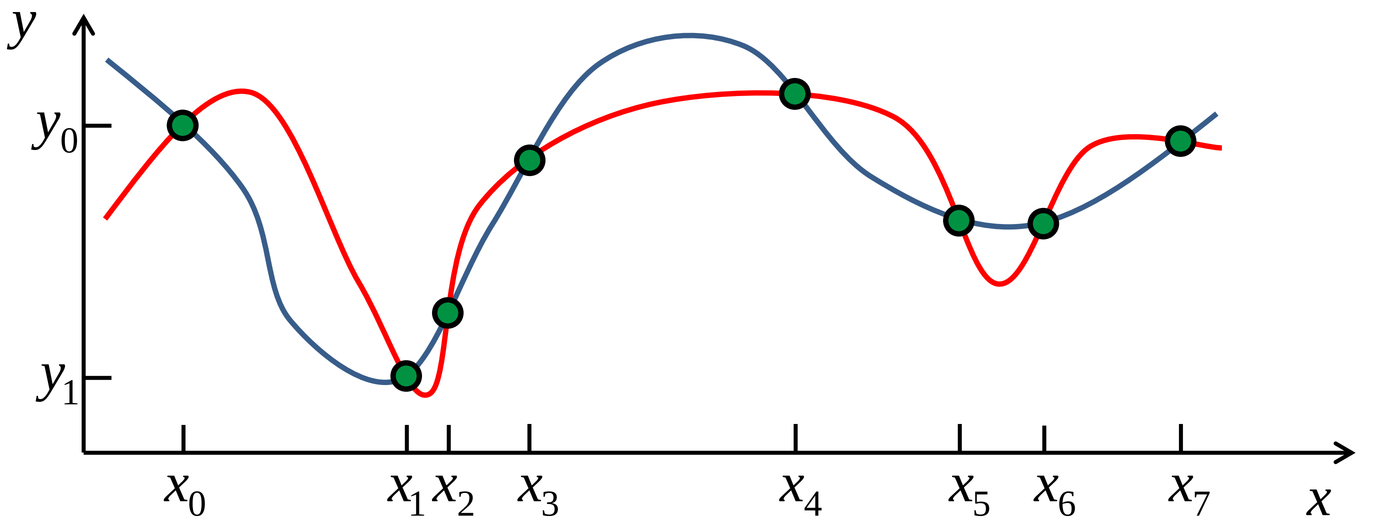
via grids and via basis sets.

Grid-point discretisation

Represent function $y(x)$ and its derivatives $\dfrac{dy}{dx}, \quad \dfrac{d^2y}{dx^2}\ldots$ using

only values of y at a discrete set of points $x_0, x_1, x_2 \ldots \equiv \{x_i\}$

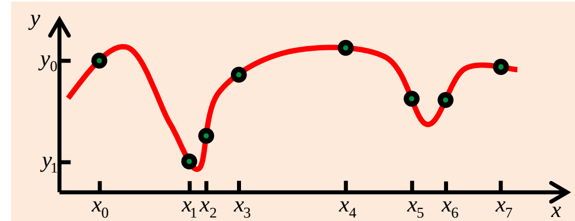
(a) Functions



Continuous $y(x)$ represented by $y(x_0), y(x_1), y(x_2)\ldots$;
write $y(x_i) = y_i$, and the set of discrete values as $\{y_i\}$.

Many functions have same $\{y_i\}$, so issue with accuracy.

(b) Derivatives



Use Taylor series

$$y(x+a) = y(x) + a y'(x) + \frac{a^2}{2} y''(x) + \ldots + \frac{a^n}{n!} y^{(n)}(x) + \ldots \qquad ①$$

to approximate derivatives at grid points, using differences between the y_i

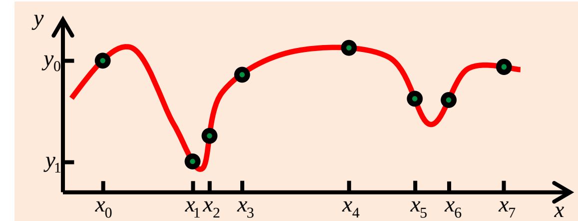
eg $y'(x)$: At grid point x_i, with $x_{i+1} = x_i + a$, ① gives

$$y(x_{i+1}) = y(x_i) + a y'(x_i) + \frac{a^2}{2} y''(x_i) + \ldots$$

$$\Rightarrow y'(x_i) = \frac{1}{a} \left\{ y(x_{i+1}) - y(x_i) - \frac{a^2}{2} y''(x_i) - \ldots \right\}$$

or, in shorthand: $y'_i = \frac{1}{a} \left\{ y_{i+1} - y_i - \frac{a^2}{2} y''_i - \ldots \right\}$

$$y_i' = \frac{1}{a}\left\{ y_{i+1} - y_i - \frac{a^2}{2} y_i'' - \ldots \right\}$$



Terms from here are unknown.
OMIT them all, to yield the
"forward difference approximation" (FDA) to $y'(x_i)$:

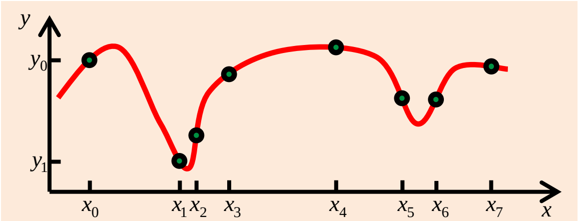
$$y_i' \approx \frac{1}{a}\left\{ y_{i+1} - y_i \right\}.$$

The discretisation error ε contains all the terms we omitted.

Here $$\varepsilon = \frac{1}{a}\left\{ \frac{a^2}{2} y_i'' + \frac{a^3}{6} y_i''' + \ldots \right\}$$

terms get smaller (if a is small)

So for small a, the first omitted term ($\propto a$) dominates ε;
we say that the FDA has discretisation error ε which is $O(a)$.



Similarly, use Taylor

$$y(x-a) = y(x) - ay'(x) + \frac{a^2}{2}y''(x) + \cdots + (-\mathbf{1})^{\boldsymbol{n}}\frac{a^n}{n!}y^{(n)}(x) + \ldots \quad ②$$

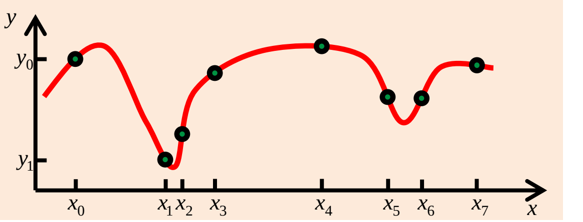
to obtain another approximation for the derivative at grid point x_i:

At grid point x_i , with $x_{i-1} = x_i - a$, ② gives

$$y(x_{i-1}) = y(x_i) - ay'(x_i) + \frac{a^2}{2}y''(x_i) + \ldots$$

$$\Rightarrow y'(x_i) = \frac{1}{a}\left\{y(x_i) - y(x_{i-1}) + \frac{a^2}{2}y''(x_i) - \ldots\right\}$$

or, in shorthand: $y_i' = \frac{1}{a}\left\{y_i - y_{i-1} + \frac{a^2}{2}y_i'' - \ldots\right\}$



$$y_i' = \frac{1}{a}\left\{y_i - y_{i-1} + \frac{a^2}{2}y_i'' - \dots\right\}$$

Again,
OMIT all terms from here, to yield the
"backward difference approximation" (BDA) to $y'(x_i)$:

$$y_i' \approx \frac{1}{a}\{y_i - y_{i-1}\}.$$

It is easy to see that BDA also has discretisation error $O(a)$.

$O(a)$ errors aren't very good. If $a \rightarrow \frac{a}{2}, \quad \varepsilon \rightarrow \frac{\varepsilon}{2}$.

Can usually do (a bit) better for not much effort.

Let us write the two versions of Taylor series side by side:

$$y(x+a) = y(x) + ay'(x) + \frac{a^2}{2}y''(x) + \cdots + \frac{a^n}{n!}y^{(n)}(x) + \ldots \qquad ①$$

$$y(x-a) = y(x) - ay'(x) + \frac{a^2}{2}y''(x) + \cdots + (-\mathbf{1})^{\boldsymbol{n}}\frac{a^n}{n!}y^{(n)}(x) + \ldots \qquad ②$$

①-② yields a "centred-difference approximation" (CDA)

$$y_i' \approx \frac{1}{2a}\left\{y_{i+1} - y_{i-1}\right\}.$$

Since the $\frac{a^2}{2}y''(x)$ terms cancel, ε is now $O(a^2)$.

Note that a **regularly-spaced grid** is a simple way to produce an $O(a^2)$ approximation to this derivative.

Let us summarize the three formulas we obtained for $y'(x_i)$:

Type	Formula	Discretisation error
y', FDA	$y_i' \approx \frac{1}{a}\{y_{i+1} - y_i\}$	$O(a)$
y', BDA	$y_i' \approx \frac{1}{a}\{y_i - y_{i-1}\}$	$O(a)$
y', CDA	$y_i' \approx \frac{1}{2a}\{y_{i+1} - y_{i-1}\}$	$O(a^2)$

Notes: 1) regular grid is required for CDA
2) FDA and BDA are useful for initial value problems
(more details later!)

What about $y''(x)$?

$$y(x+a) = y(x) + ay'(x) + \frac{a^2}{2}y''(x) + \cdots + \frac{a^n}{n!}y^{(n)}(x) + \ldots \quad (1)$$

$$y(x-a) = y(x) - ay'(x) + \frac{a^2}{2}y''(x) + \cdots + (-\mathbf{1})^{\boldsymbol{n}}\frac{a^n}{n!}y^{(n)}(x) + \ldots \quad (2)$$

(1)+(2) gives

$$y_{i+1} + y_{i-1} = 2y_i + a^2y''(x_i) + \frac{a^4}{12}y''''(x_i) + \cdots.$$

And so we obtain:

$$y''(x_i) = \frac{1}{a^2}\left[y_{i+1} + y_{i-1} - 2y_i - \frac{a^4}{12}y''''(x_i) - \cdots\right]$$

Omit terms from here.

Easy to see that the error is $O(a^2)$

A useful set of formulas for discrete derivatives:

Type	Formula	Error
y', FDA	$y_i' \approx \frac{1}{a}\{y_{i+1} - y_i\}$	$O(a)$
y', BDA	$y_i' \approx \frac{1}{a}\{y_i - y_{i-1}\}$	$O(a)$
y', CDA	$y_i' \approx \frac{1}{2a}\{y_{i+1} - y_{i-1}\}$	$O(a^2)$
y'', CDA	$y''(x_i) \approx \frac{1}{a^2}[y_{i+1} + y_{i-1} - 2y_i]$	$O(a^2)$

Note: regular grid is required for CDA formulas!

- *A side note:* One could attempt to obtain $y''(x_i)$ by applying recursively the earlier derived formulas for y':

$$y''(x) = \frac{d}{dx}[y'(x)] \quad \Rightarrow \quad y''(x_i) \approx \frac{1}{2a}[y'(x_{i+1}) - y'(x_{i-1})]$$

with $$y'(x_i) \approx \frac{1}{2a}[y(x_{i+1}) - y(x_{i-1})]$$

It is easy to see [check it yourself!] that this is equivalent to

$$y''(x_i) \approx \frac{1}{4a^2}[y_{i+2} + y_{i-2} - 2y_i]$$

I.e. we obtained exactly the same formula as we derived earlier, only with a twice larger discretization step $a \to 2a$

Hence the error is 4 times larger - do not do this!

Computational cost is the reason to chase $\varepsilon \sim O(a^2)$

Express in terms of required memory M and time T

Will see later that to solve a discretised problem with N grid points

$$M \sim N^2 \qquad T \sim N^3$$

So for an ODE with fixed domain and regular grids,
halving the grid spacing $a \to a/2$ requires $N \to 2N$ points
leading to $\boldsymbol{M \to 4M}$ and $\boldsymbol{T \to 8T}$

- an expensive change if it only halves ϵ !

- $\varepsilon \sim O(a^2)$ is not great, but MUCH better than $\varepsilon \sim O(a)$

- There are many other discretisation approximations for various order derivatives – check literature. Different problems require different approaches.

- The four formulas we derived (FDA, BDA, CDA for y', and CDA for y'') are the most common.

- $O(a^3)$ errors are achievable, but require much more effort – often not practical.

An example: discretize the following Boundary Value Problem

$$y' + y^4 = 1$$

with Dirichlet conditions $y(0) = 0, y(1) = 3$

Summary:

- Grid-point discretization is at the core of numerical modelling of many ODE and PDE problems.

- There are many approximations for derivatives via finite differences. Most common are the central-difference based involving nearest neighbours, but other options are also available.

- Always need to consider the cost (in terms of memory and computation time) required to reduce the computation error.

- Solving ODEs and PDEs with grid-point discretization is generally known as the **Finite Difference Method**

Lecture 5:

Basis-set discretisation

Basis-set discretisation

Expand functions as a sum of BASIS FUNCTIONS.

$$f(t) = \sum_n c_n F_n(t) = c_1 F_1(t) + c_2 F_2(t) + c_3 F_3(t) + \cdots$$

- The sum is formally infinite, but will have to be truncated in numerics

- The (discrete) set of coefficients c_n describes the function

- BASIS functions are some known analytical functions.

Some popular choices are:

- Fourier series expansion *(periodic functions)*
- Hermite polynomials *(quantum harmonic oscillator type equations)*
- Hankel functions *(dispersion/diffraction in radially symmetric geometries)*

Basis-set discretisation

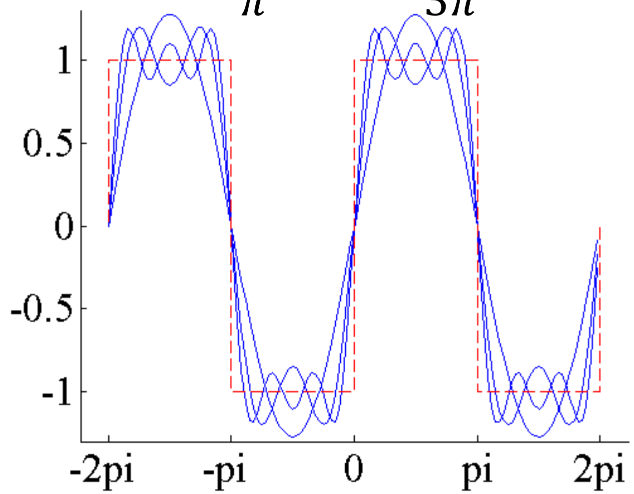
Expand functions as a sum of BASIS FUNCTIONS.

An example: Fourier series

$$f(t)=\sum_{n=-\infty}^{\infty}c_n e^{in\omega t};\quad g(t)=\frac{a_0}{2}+\sum_{n=1}^{\infty}\left[a_n\cos(n\omega t)+b_n\sin(n\omega t)\right]$$

$f(t)$ must be periodic! $f(t+T)=f(t)\ \forall\ t,\ T=2\pi/\omega$

For a square wave $g(t)=\frac{4}{\pi}\sin t+\frac{4}{3\pi}\sin 3t+\frac{4}{5\pi}\sin 5t+\ldots$



$$T=2\pi;\quad \omega=1$$

Note that $g(t)$ is defined for ALL t.

Discretisation error (usually) arises once the sum is truncated.

Basis-set discretisation

How to choose a basis set?

$$f(t) = \sum_n c_n F_n(t) = c_1 F_1(t) + c_2 F_2(t) + c_3 F_3(t) + \cdots$$

Main criteria when selecting the set $F_n(t)$:

- (PREFERABLY) a COMPLETE and ORTHONORMAL set

(means you can expand ANY function)

(this is useful for deriving equations for c_n - see further examples)

- Consistency with the auxiliary conditions

[e.g. complex exponents $F_n(t) = \exp(in\omega t)$ satisfy periodic boundary conditions: $F_n\left(t + \frac{2\pi}{\omega}\right) = F_n(t)$]

- Speed of numerical conversion from/to the basis set

(i.e. for any function $f(t)$ you should be able to obtain the corresponding set of coefficients c_n, and the same in the opposite direction)

Differential operators act on the individual basis functions.

eg $$f(t) = \sum_n c_n \mathrm{e}^{in\omega t}\,; \quad \frac{df}{dt} = \sum_n in\omega c_n \mathrm{e}^{in\omega t}$$

So, to discretise an ODE via basis sets:

- Choose a set of basis functions
- Substitute expansion into ODE and re-write as an algebraic equation in the expansion coefficients (c_n)
- Truncate summation if computing…
- Make sure the auxiliary conditions are satisfied (can lead to additional conditions on c_n)

Complex Fourier series expansion

$$f(t) = \sum_{n=-\infty}^{\infty} c_n \mathrm{e}^{in\omega_0 t}$$

- Eigen-functions of differential operators

$$\frac{d^n}{dt^n}\exp(in\omega_0 t) = (in\omega_0)^n \exp(in\omega_0 t)$$

- Requires periodic boundary condition!

$$f(t+T) = f(t) \qquad\qquad \omega_0 = 2\pi/T$$

- The set is complete, and orthonormal:

$$e_l(t) = e^{il\omega_0 t}$$
$$e_m(t) = e^{im\omega_0 t}$$

$$\Rightarrow \frac{1}{T}\int_0^T e_l^*(t) e_m(t) dt = \frac{1}{T}\int_0^T e^{i(m-l)\omega_0 t} dt = \boxed{\delta_{l,m}}$$

Kronecker delta:

$$\delta_{l,m} = \begin{cases} 0, if\ l \neq m \\ 1, if\ l = m \end{cases}$$

(revise Fourier series course!)

An example (Boundary Value Problem):

$$\frac{d^2\Phi}{dx^2} = x, \qquad \Phi(-L/2) = 0, \qquad \Phi(L/2) = 0.$$

Actually, we know the analytical solution :

$$\Phi(x) = \frac{1}{6}x\left(x^2 - \frac{L^2}{4}\right),$$

But we will pretend we do not know it,

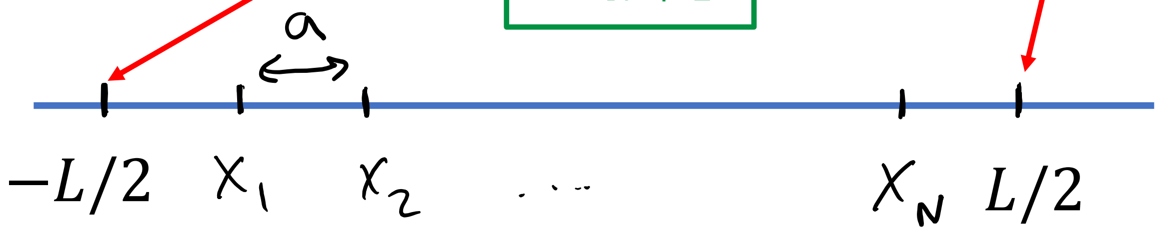
and **try to find it numerically** with point-grid and basis-set discretisations

An example (Boundary Value Problem):

$$\frac{d^2\Phi}{dx^2} = x, \qquad \Phi(-L/2) = 0, \qquad \Phi(L/2) = 0.$$

- <u>Grid discretisation</u>:

$$\boxed{a = \frac{L}{N+1}}$$



$$\frac{1}{a^2}\left(\phi_{j+1} + \phi_{j-1} - 2\phi_j\right) = x_j = -\frac{L}{2} + j\frac{L}{N+1}, \qquad j = 2,3,\ldots N-1$$

$$\frac{1}{a^2}\left(\phi_2 + \mathbf{0} - 2\phi_1\right) = x_1 = -\frac{L}{2} + \frac{L}{N+1}, \qquad j = 1$$

$$\frac{1}{a^2}\left(\mathbf{0} + \phi_{N-1} - 2\phi_N\right) = x_N = -\frac{L}{2} + N\frac{L}{N+1}, \qquad j = N$$

An example (Boundary Value Problem):

$$\frac{d^2\Phi}{dx^2} = x, \qquad \Phi(-L/2) = 0, \qquad \Phi(L/2) = 0.$$

- Grid discretisation ➡ linear matrix equation

$$\boxed{\hat{M}\vec{x} = \vec{b},} \qquad \vec{x} = [\phi_1, \phi_2, \dots, \phi_N]^T,$$ **- unknowns**

$$\hat{M} = \frac{1}{a^2}\begin{bmatrix} -2 & 1 & 0 & \cdots & \cdots & \cdots & 0 \\ 1 & -2 & 1 & 0 & \cdots & \cdots & 0 \\ 0 & 1 & -2 & 1 & 0 & \cdots & 0 \\ \cdots & \cdots & \ddots & \ddots & \ddots & \cdots & \cdots \\ 0 & \cdots & 0 & 1 & -2 & 1 & 0 \\ 0 & \cdots & \cdots & 0 & 1 & -2 & 1 \\ 0 & \cdots & \cdots & \cdots & 0 & 1 & -2 \end{bmatrix}$$

- $N \times N$ tri-diagonal matrix

$$\vec{b} = \left[-\frac{L}{2} + \frac{L}{N+1}, -\frac{L}{2} + \frac{2L}{N+1}, \dots, -\frac{L}{2} + \frac{NL}{N+1}\right]^T,$$

An important observation so far:

Our initial problem is a differential equation

$$\frac{d^2\Phi}{dx^2} = x, \qquad \Phi(-L/2) = 0, \qquad \Phi(L/2) = 0.$$

By implementing grid-point discretization, we reduced it to the linear algebra problem (a set of linear algebraic equations):

$$\boxed{\hat{M}\vec{x} = \vec{b},}$$

for the N-component vector of unknowns $\vec{x}$ which gives N values of the (discretized) function $\Phi(x)$ at regular grid points in the interval $(-L/2, L/2)$

This linear algebra problem can be solved on a computer
(we will learn later how!)

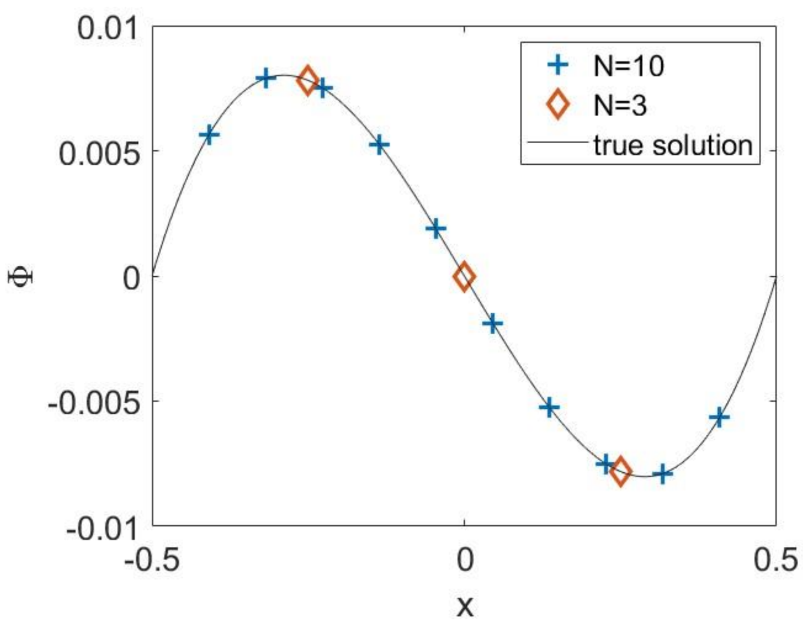
- **Grid discretisation** ➡ linear matrix equation

$$\boxed{\widehat{M}\vec{x} = \vec{b},}$$

$$\vec{x} = [\phi_1, \phi_2, \dots, \phi_N]^T,$$ **- unknowns**

$$\vec{b} = [a, 2a, 3a, \dots, Na\,]^T,$$

Solve the matrix equation numerically (e.g. in Matlab):



See "test_grid_point_vs_basis_set.m" matlab code uploaded on Moodle page

An example (Boundary Value Problem):

$$\frac{d^2\Phi}{dx^2} = x, \qquad \Phi(-L/2) = 0, \qquad \Phi(L/2) = 0.$$

- Basis set

$$\Phi(x) = \sum_{n=-\infty}^{\infty} c_n \mathrm{e}^{in\omega_0 x} \qquad \text{select} \qquad \omega_0 = 2\pi/L$$

Note: this means $\Phi(x + L) = \Phi(x)$

Compatible with the boundary conditions??

Dirichlet BC sets the value for $\Phi(0)$ – this is more restricting than periodic BC!

Substitute into the equation:

$$-\sum_n (n\omega_0)^2 c_n e^{in\omega_0 x} = x$$

$$\frac{d^2\Phi}{dx^2} = x, \qquad \Phi(-L/2) = 0, \qquad \Phi(L/2) = 0$$

$$\Phi(x) = \sum_{n=-\infty}^{\infty} c_n \mathrm{e}^{in\omega_0 x}, \qquad \omega_0 = 2\pi/L$$

$$-\sum_n (n\omega_0)^2 c_n e^{in\omega_0 x} = x$$

Apply "closure": multiply both sides by $\exp(-im\omega_0 x)$ and integrate over the period:

$$-\sum_n (n\omega_0)^2 c_n \underbrace{\frac{1}{L}\int_0^L \exp[i(n-m)\omega_0 x]dx}_{=\delta_{n,m}} = \underbrace{\frac{1}{L}\int_0^L x \exp[-im\omega_0 x]dx}_{=F_m}$$

$= \delta_{n,m}$ **(orthogonality)**

$= F_m$ **(Fourier series coefficient of *[the periodic version]* of $f(x) = x$)**

$$\boxed{(m\omega_0)^2 c_m = -F_m = \frac{i(-1)^m}{m\omega_0}}$$

Note: *in this example the integral is easy to tackle analytically and obtain this result for F_m. In real-life situation, you will need to compute such integrals numerically. We will learn how to do it later! (Fast Fourier Transforms)*

$$\frac{d^2\Phi}{dx^2} = x, \qquad \boxed{\Phi(-L/2) = 0, \qquad \Phi(L/2) = 0}$$

$$\Phi(x) = \sum_{n=-\infty}^{\infty} c_n e^{in\omega_0 x}, \qquad \omega_0 = 2\pi/L$$

$$(m\omega_0)^2 c_m = -F_m = \frac{i(-1)^{m+1}}{m\omega_0}$$

In this case, we obtained a trivial set of equations for c_m coefficients, which we can solve by hand:

$$\Rightarrow \quad \boxed{c_m = \frac{i(-1)^{m+1}}{(m\omega_0)^3}}$$

- **But what about c_0??** $\quad (0 \cdot \omega_0)^2 c_0 = -F_0$
- It is easy to see that $F_0 = \frac{1}{L}\int_0^L x \exp[-i(0 \cdot \omega_0)x]dx = 0$
- Does that mean that c_0 can be anything??

  No! Need to satisfy Auxiliary Conditions!

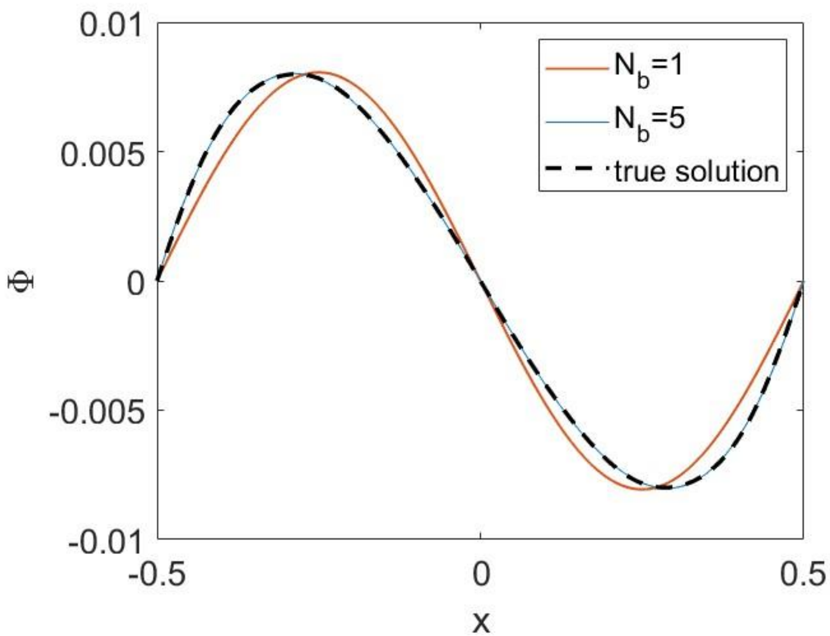
  $\Rightarrow$ Need to set $c_0 = 0$

$$\frac{d^2\Phi}{dx^2} = x, \qquad \Phi(-L/2) = 0, \qquad \Phi(L/2) = 0$$

- **Basis-set**

$$\Phi(x) = \sum_{\substack{n=-\infty,\\ n\neq 0}}^{\infty} \frac{i(-1)^{m+1}}{(m\omega_0)^3} \mathrm{e}^{in2\pi x},$$

In practice, you will need to truncate this sum at some $n = \pm N_b$



See "test_grid_point_vs_basis_set.m" matlab code uploaded on Moodle page

- **Original problem: linear ODE boundary value problem**

$$\frac{d^2\Phi}{dx^2} = x, \qquad \Phi(-L/2) = 0, \qquad \Phi(L/2) = 0$$

1. Discretize

Grid: $\phi_j = \Phi(\mathrm{x_j})$

E.g. for N=3 points:

$$\frac{1}{(L/4)^2}(\phi_3 + \phi_1 - 2\phi_2) = 0,$$

$$\frac{1}{(L/4)^2}(\phi_2 - 2\phi_3) = \frac{L}{4},$$

$$\frac{1}{(L/4)^2}(\phi_2 - 2\phi_1) = -\frac{L}{4},$$

Basis-set: $\Phi(\mathrm{x}) = \sum_n c_n \exp\left(in\left(\frac{2\pi}{L}\right)t\right)$

Truncate infinite sum e.g. for $-1 \le n \le 1$

$$\left(-\frac{2\pi}{L}\right)^2 c_{-1} = -\frac{iL}{2\pi}$$

$$c_0 = 0$$

$$\left(\frac{2\pi}{L}\right)^2 c_1 = \frac{iL}{2\pi}$$

2. Solve (on a computer) **the set of linear algebraic equations**

3. Obtain values of the function at the grid points $\boldsymbol{\phi_j}$

3. Obtain values of the expansion coefficients $\boldsymbol{c_n}$

Summary:

Grid

- Easy to setup, including various auxiliary conditions

- Reduces the problem to the set of N coupled algebraic equations

(in higher-dimensional problems the number of equations grows exponentially!)

- **Solution is obtained at discrete points => may be problematic for post-processing**

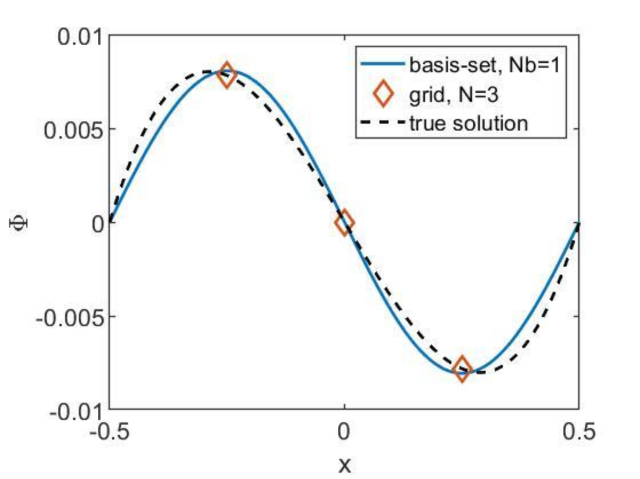
Basis-set

- **Requires some effort to setup, take special care of:**
  - *auxiliary conditions;*
  - *choice of the basis set*

- **Reduces the problem to the set of N coupled algebraic equations**

(with a proper basis set choice) **easy to solve and/or have good convergence** *(i.e. require only few equations to solve)*

- **Solution is obtained in analytic form: convenient for post-processing BUT can have poor convergence**



Lecture 6:

Discretisation: ODEs with variable coefficients and nonlinear ODEs

An example: find localized states in a 1D quantum well

$$-\frac{\hbar^2}{2m}\frac{d^2\Psi}{dx^2} + \boldsymbol{V(x)}\Psi = E\Psi,$$

$$u(\xi) = \frac{V(\xi)}{E_0}$$

1) De-dimensionalize:

$$x = \frac{\hbar}{\sqrt{2mE_0}}\xi$$

$$E = E_0\epsilon$$

$$\boxed{-\frac{d^2\Psi}{d\xi^2} + \boldsymbol{u(\xi)}\Psi = \epsilon\Psi,}$$

E_0 - a "characteristic" energy (can choose a value to rescale the potential function $u = V/E_0$)

2) Interested in localized states only => can have freedom in choosing boundary conditions for **large enough computational window**

Note: how large is "large"? Requires some physical intuition!

3) Attempt point-grid and basis-set (Fourier) discretisations

ODE with a variable coefficient

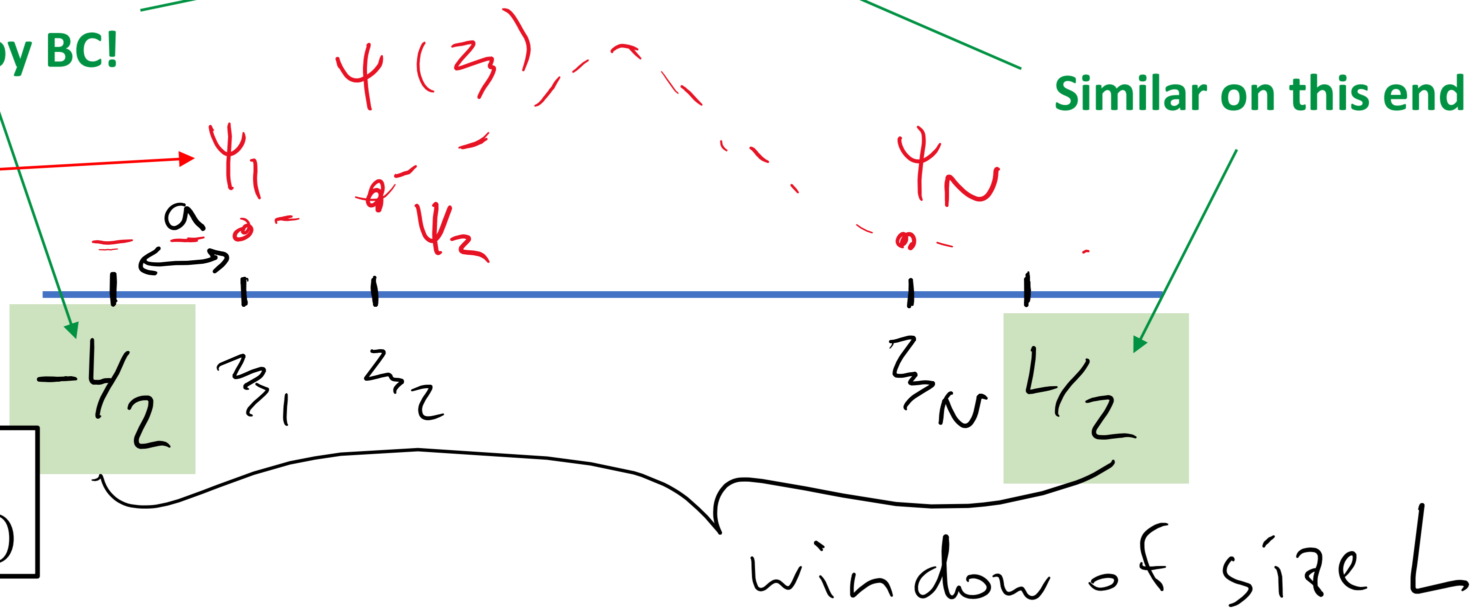
Grid (v1): **Dirichlet BCs:** $\Psi(-L/2) = 0,\ \Psi(L/2) = 0$

$$-\frac{d^2\Psi}{d\xi^2} + u(\xi)\Psi = \epsilon\Psi,$$

$\Psi\left(-\frac{L}{2}\right)$ is set by BC!

Similar on this end

the grid starts at $\xi = -\frac{L}{2} + a$



Grid step
$\boldsymbol{a = L/(N+1)}$

$$-\frac{1}{a^2}(\Psi_{j+1} + \Psi_{j-1} - 2\Psi_j) + u_j\Psi_j = \epsilon\Psi_\mathrm{j}, \qquad j = 2,3, \dots N-1$$

Dirichlet BCs:

$$-\frac{1}{a^2}(\Psi_2 + \mathbf{0} - 2\Psi_1) + u_1\Psi_1 = \epsilon\Psi_1, \qquad j = 1$$

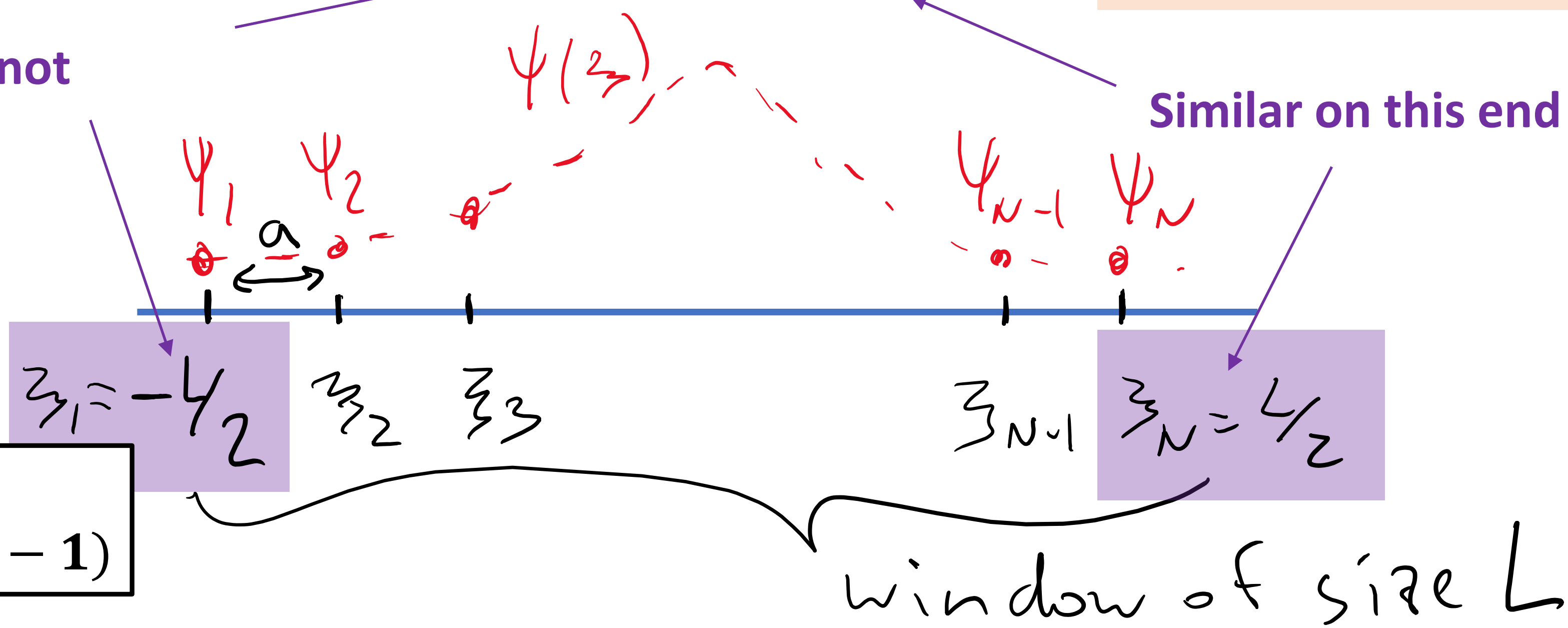
$$-\frac{1}{a^2}(\mathbf{0} + \Psi_{N-1} - 2\Psi_N) + u_N\Psi_N = \epsilon\Psi_\mathrm{N}, \qquad j = N$$

Grid (v2): **Neumann BCs:** $\Psi'(-L/2) = 0,\ \Psi'(L/2) = 0$

$$-\frac{d^2\Psi}{d\xi^2} + u(\xi)\Psi = \epsilon\Psi,$$

$\Psi\left(-\frac{L}{2}\right)$ is not known!

Similar on this end



Grid step
$\boldsymbol{a = L/(N-1)}$

$$-\frac{1}{a^2}(\Psi_{j+1} + \Psi_{j-1} - 2\Psi_j) + u_j\Psi_j = \epsilon\Psi_j, \qquad j = 2,3,\ldots N-1$$

Apply BCs V2a: Use FDA/BDA

$$-\frac{1}{a^2}(\Psi_2 + \boldsymbol{\Psi_1} - 2\Psi_1) + u_1\Psi_1 = \epsilon\Psi_1, \quad j = 1 \qquad a\Psi_1' = (\Psi_1 - \Psi_0)$$

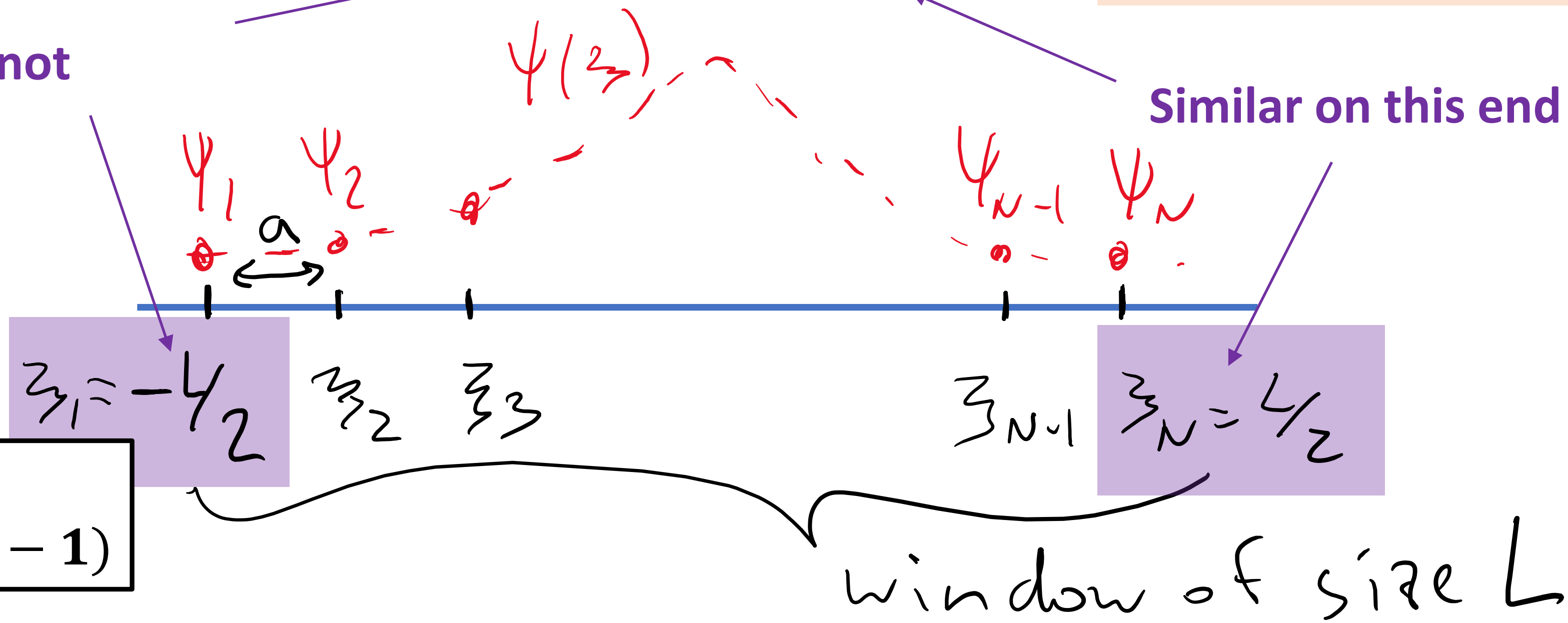
$$-\frac{1}{a^2}(\boldsymbol{\Psi_N} + \Psi_{N-1} - 2\Psi_N) + u_N\Psi_N = \epsilon\Psi_N, \quad j = N \qquad a\Psi_N' = (\Psi_{N+1} - \Psi_N)$$

Grid (v2): **Neumann BCs:** $\Psi'(-L/2) = 0,\ \Psi'(L/2) = 0$

$$-\frac{d^2\Psi}{d\xi^2} + u(\xi)\Psi = \epsilon\Psi,$$

$\Psi\left(-\frac{L}{2}\right)$ is not known!

Similar on this end



Grid step
$\boldsymbol{a = L/(N-1)}$

$$-\frac{1}{a^2}(\Psi_{j+1} + \Psi_{j-1} - 2\Psi_j) + u_j\Psi_j = \epsilon\Psi_{\mathrm{j}}, \qquad j = 2,3,\dots N-1$$

Apply BCs V2b: Use CDA

$$-\frac{1}{a^2}(\Psi_2 + \boldsymbol{\Psi_2} - 2\Psi_1) + u_1\Psi_1 = \epsilon\Psi_1, \quad j = 1 \qquad \Psi_1' = \frac{\Psi_2 - \Psi_0}{2a}$$

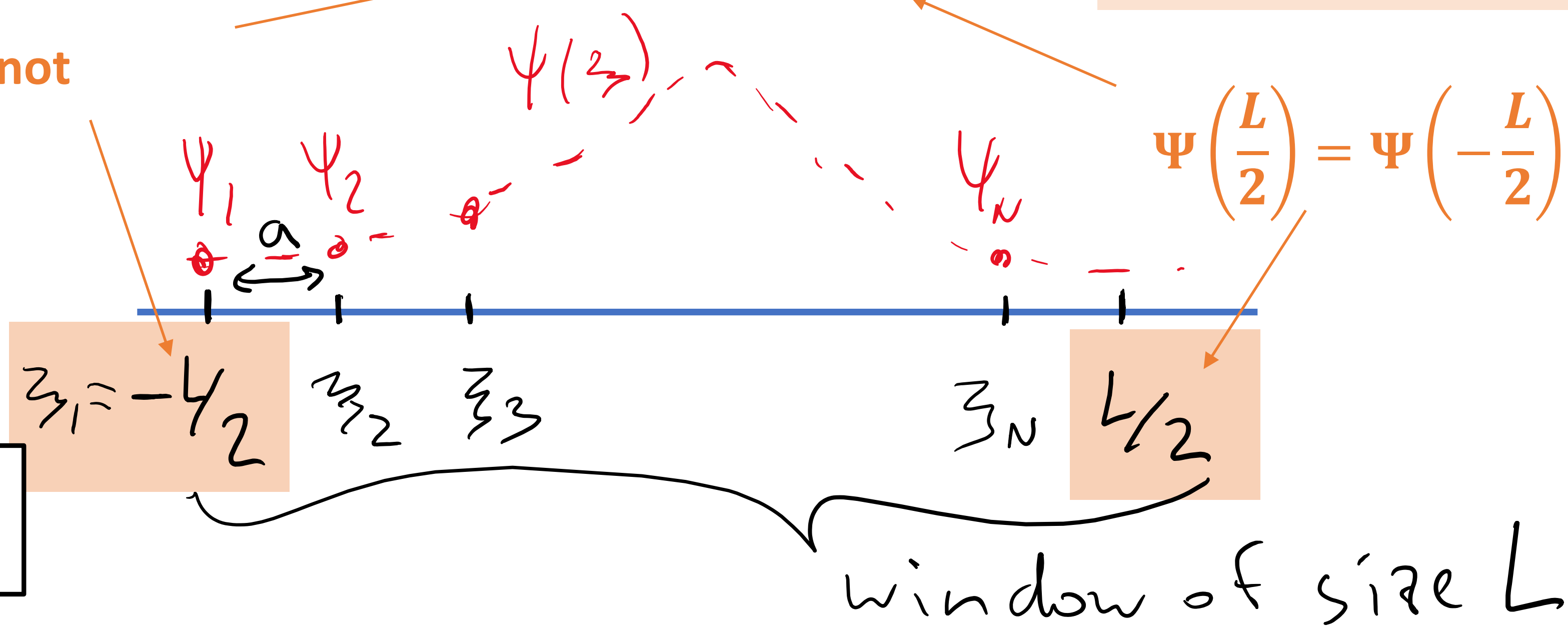
$$-\frac{1}{a^2}(\boldsymbol{\Psi_{N-1}} + \Psi_{N-1} - 2\Psi_N) + u_N\Psi_N = \epsilon\Psi_{\mathrm{N}}, \ j = N \qquad \Psi_N' = \frac{\Psi_{N+1} - \Psi_{N-1}}{2a}$$

Grid (v3): **Periodic BCs:** $\Psi(x+L)=\Psi(x)$

$$-\frac{d^2\Psi}{d\xi^2}+u(\xi)\Psi=\epsilon\Psi,$$

$\Psi\left(-\frac{L}{2}\right)$ is not known!

$$\boldsymbol{\Psi\left(\frac{L}{2}\right)=\Psi\left(-\frac{L}{2}\right)}$$



Grid step
$\boldsymbol{a=L/N}$

$$-\frac{1}{a^2}(\Psi_{j+1}+\Psi_{j-1}-2\Psi_j)+u_j\Psi_j=\epsilon\Psi_{\rm j},\qquad j=2,3,\ldots N-1$$

Apply BCs:

$$-\frac{1}{a^2}(\Psi_2+\boldsymbol{\Psi_N}-2\Psi_1)+u_1\Psi_1=\epsilon\Psi_1,\quad j=1$$

$$-\frac{1}{a^2}(\boldsymbol{\Psi_1}+\Psi_{N-1}-2\Psi_N)+u_N\Psi_N=\epsilon\Psi_{\rm N},\qquad j=N$$

$$-\frac{d^2\Psi}{d\xi^2} + u(\xi)\Psi = \epsilon\Psi,$$

Grid:

- **Eigen-value problem**

$$\boxed{\widehat{M}\vec{x} = \epsilon\vec{x},}$$

$\vec{x} = [\Psi_1, \Psi_2, \dots, \Psi_N]^T$ - **Unknown discretized wave function** ***(eigen-vector)***

ϵ - **Unknown energy *(eigen-value)***

Note: expect real values for ϵ, but complex $\vec{x}$

$$\widehat{M} = \frac{1}{a^2}\begin{bmatrix} A + a^2u_1 & B & 0 & \cdots & \cdots & \cdots & C \\ -1 & 2 + a^2u_2 & -1 & 0 & \cdots & \cdots & 0 \\ 0 & -1 & \ddots & -1 & 0 & \cdots & 0 \\ \cdots & \cdots & \ddots & \ddots & \ddots & \cdots & \cdots \\ 0 & \cdots & 0 & -1 & \ddots & -1 & 0 \\ 0 & \cdots & \cdots & 0 & -1 & \ddots & -1 \\ C & \cdots & \cdots & \cdots & 0 & B & A + a^2u_N \end{bmatrix}$$

- *$N \times N$* ***sparce matrix***
  - *coeffs A,B,C are determined by BCs*
  - *real matrix*
  - *symmetric*
  - *Tri-diagonal, unless using periodic BCs*

Grid:

$$-\frac{d^2\Psi}{d\xi^2} + u(\xi)\Psi = \epsilon\Psi,$$

- **Eigen-value problem**

$$\boxed{\hat{M}\vec{x} = \epsilon\vec{x},}$$

- **Use numerical algorithms to solve the eigen-value problem**

  ➡ *Can try to utilize an optimized method for the specific properties of* $\hat{M}$

- **Expect N eigenvalues (and eigenvectors)**

- **How to identify *LOCALIZED* eigenvectors?**

  ➡ Use your physics knowledge!

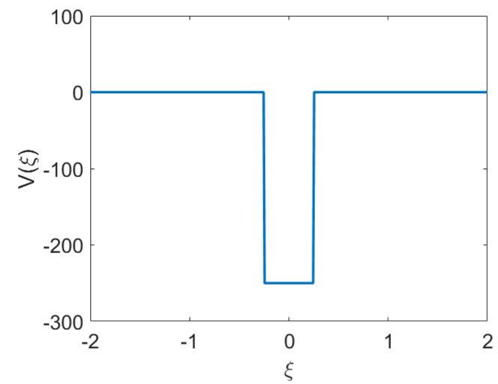
E.g. expect negative energies for localized states

Grid:

$$-\frac{d^2\Psi}{d\xi^2} + u(\xi)\Psi = \epsilon\Psi,$$

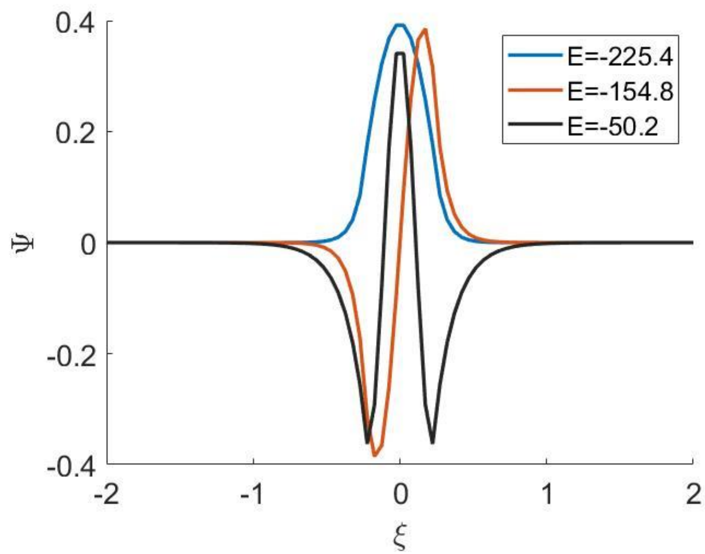
- **Eigen-value problem**

$$\boxed{\widehat{M}\vec{x} = \epsilon\vec{x},}$$

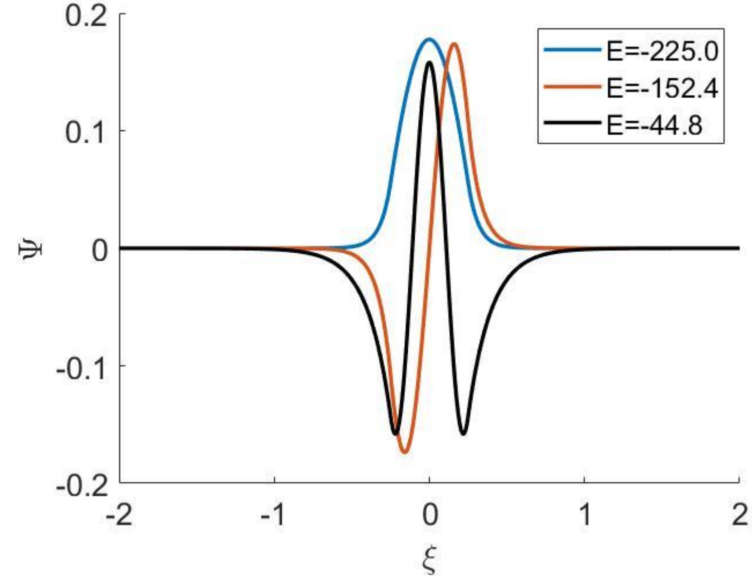


- Numerically obtained eigen-states in a square well potential

$N = 100$



$N = 500$



See "Schroedinger_ev_point_grid.m" matlab code uploaded on Moodle page

A useful shortcut

$$-\frac{d^2\Psi}{d\xi^2} + u(\xi)\Psi = \epsilon\Psi,$$

- **Eigen-states of a *linear* *(!)* symmetric problem inherit the symmetry (are either odd- or even-)**

Note: This is only true for linear problems

- **Could solve on half-interval with the symmetry conditions imposed at $x = 0$:**

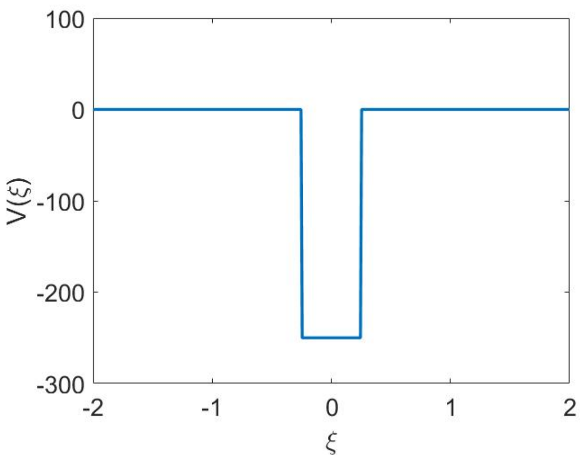
$$\Psi(-x) = \Psi(x)$$

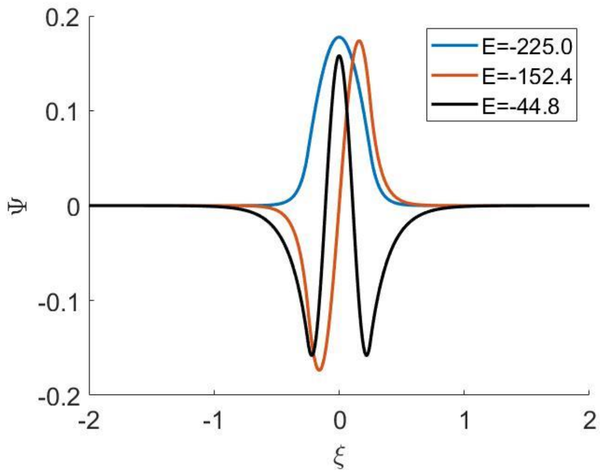
AND $$\Psi(-x) = -\Psi(x)$$

- Reduce the number of grid points by half, but need to solve twice. Do you gain anything?

- Remember for matrix problems such as eigenvalue memory scales as $\mathrm{M} \sim N^2$ and time scales as $T \sim N^3$

- **YES, the gain is worth the effort, especially for large grids**





Basis-set *(using Fourier Series)*:

$$-\frac{d^2\Psi}{d\xi^2} + u(\xi)\Psi = \epsilon\Psi,$$

$$\Psi(\xi) = \sum_{n=-\infty}^{\infty} c_n \mathrm{e}^{in\omega_0\xi}, \qquad \omega_0 = \frac{2\pi}{L}$$

Note: *Using Fourier series expansion =>*
assume periodic boundary conditions

- **Substitute:**

$$-\textstyle\sum_{n=-\infty}^{\infty}(-n\omega_0)^2 c_n \mathrm{e}^{in\omega_0\xi} + u(\xi)\cdot\sum_{n=-\infty}^{\infty} c_n \mathrm{e}^{in\omega_0\xi} = \epsilon\cdot\sum_{n=-\infty}^{\infty} c_n \mathrm{e}^{in\omega_0\xi}$$

- **Apply "closure"** (multiply by $e^{-im\omega_0\xi}$ and integrate $\left(\frac{1}{L}\right)\int_0^L \ldots d\xi$):

$$-\sum_{n=-\infty}^{\infty}(-n\omega_0)^2 c_n \delta_{n,m} + \frac{1}{L}\int_0^L u(\xi)\cdot\sum_{n=-\infty}^{\infty} c_n \mathrm{e}^{i(n-m)\omega_0\xi}\, d\xi = \epsilon\cdot\sum_{n=-\infty}^{\infty} c_n \delta_{n,m}$$

Basis-set *(using Fourier Series)*:

$$-\frac{d^2\Psi}{d\xi^2} + u(\xi)\Psi = \epsilon\Psi,$$

$$-\sum_{n=-\infty}^{\infty}(-n\omega_0)^2 c_n\delta_{n,m} + \frac{1}{L}\int_0^L u(\xi)\cdot\sum_{n=-\infty}^{\infty} c_n \mathrm{e}^{i(n-m)\omega_0\xi}\, d\xi = \epsilon\cdot\sum_{n=-\infty}^{\infty} c_n\delta_{n,m}$$

$$m^2\omega_0^2 c_m + \sum_{n=-\infty}^{\infty} U_{m-n}\cdot c_n = \epsilon\cdot c_m$$

where

$$U_{m-n} = \frac{1}{L}\int_0^L u(\xi)e^{-i(m-n)\omega_0\xi}d\xi$$

is $(m-n)$th complex **Fourier Series coefficient** of the *(periodic version of)* potential function $u(\xi)$

Basis-set *(using Fourier Series)*:

$$-\frac{d^2\Psi}{d\xi^2} + u(\xi)\Psi = \epsilon\Psi,$$

Can write it in the matrix form:

$$\Psi(\xi) \approx \sum_{n=-N}^{N} c_n e^{in(2\pi/L)\xi},$$

$$\boxed{\hat{M}\vec{c} = \epsilon\vec{c},}$$

$\vec{c} = [c_{-N}, c_{-N+1}, \dots, c_0, c_1, \dots, c_N]^T$ - **Unknown Fourier coeffs** ***(eigen-vector)***

ϵ - **Unknown energy *(eigen-value)***

$$\hat{M} = \begin{bmatrix} D_{-N} & U_{-1} & U_{-2} & \cdots & \cdots & \cdots & U_{-2N} \\ U_1 & D_{-N+1} & U_{-1} & U_{-2} & \cdots & \cdots & U_{-2N+1} \\ U_2 & U_1 & \ddots & \ddots & \ddots & \cdots & \cdots \\ \cdots & \cdots & U_1 & D_0 & U_{-1} & \cdots & \cdots \\ \cdots & \cdots & \dots & \ddots & D_1 & \ddots & \dots \\ \cdots & \cdots & \cdots & \dots & \ddots & \ddots & U_{-1} \\ U_{2N} & \cdots & \cdots & \cdots & U_2 & U_1 & D_N \end{bmatrix}$$

- $(2N+1) \times (2N+1)$ ***full matrix***

- *Real* $u(\xi)$ *=>* $U_k = U_{-k}^*$

  => Hermitian matrix

- *if* $u(-\xi) = u(\xi)$ *=>* $U_k = U_{-k}$

  => symmetric matrix

$$D_k = k^2\omega_0^2 + U_0 \qquad U_k = \frac{1}{L}\int_0^L u(\xi) e^{-ik\omega_0\xi} d\xi$$

Basis-set vs grid

$$-\frac{d^2\Psi}{d\xi^2} + u(\xi)\Psi = \epsilon\Psi,$$

- **For a comparable matrix size, basis-set approach will take much longer to solve**

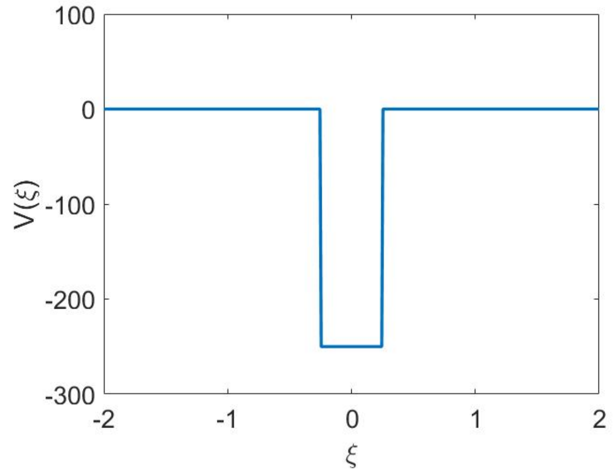
- **Discretization error in basis-set is due to truncation in the Fourier series. The required number of harmonics depends on the problem**

$$\Psi(\xi) \approx \sum_{n=-N}^{N} c_n \mathrm{e}^{in(2\pi/L)\xi},$$

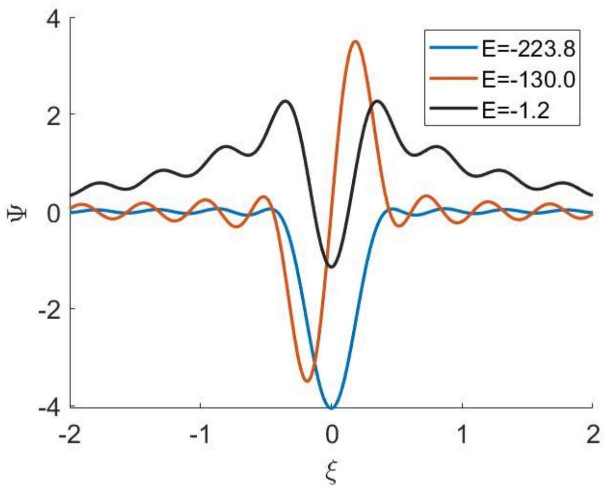
- **Basis-set could still be advantageous, if the Fourier series convergence is good**

Basis-set: square well potential example

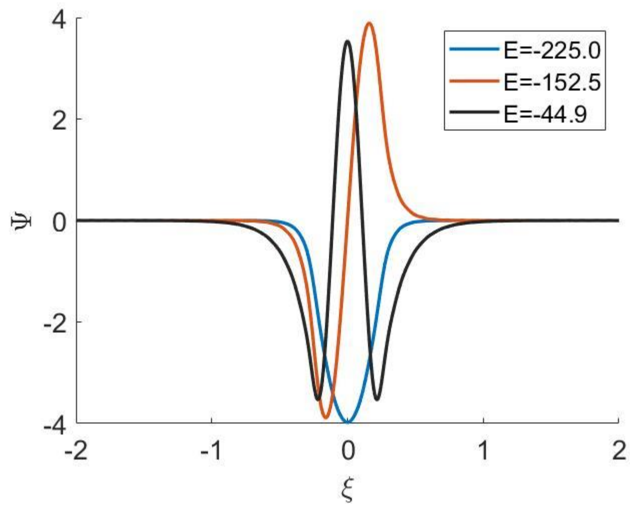
$$-\frac{d^2\Psi}{d\xi^2} + u(\xi)\Psi = \epsilon\Psi,$$



$N = 10$



$N = 50$



See "Schroedinger_ev_basis_set.m" matlab code uploaded on Moodle page

Another example: nonlinear pendulum

$$\ddot{\theta} + \frac{g}{l}\sin\theta = 0, \qquad \theta(0) = 0, \qquad \theta(T) = 0$$

1) De-dimensionalize:

$$t = \sqrt{l/g}\,\tau \quad \Rightarrow \quad \boxed{\frac{d^2\theta}{d\tau^2} + \sin\theta = 0, \qquad \theta(0) = 0, \theta(\tau_0) = 0}$$

$$T_0 = \sqrt{l/g}$$

- ***Period of the linear pendulum*** *(i.e. in the limit $\theta \ll 1$)*

$$\tau_0 = T/T_0$$

- ***Dimensionless period***

Grid:

$$\frac{d^2\theta}{d\tau^2} + \sin\theta = 0, \qquad \theta(0) = 0, \theta(\tau_0) = 0$$

$$\frac{1}{a^2}(\theta_{j+1} + \theta_{j-1} - 2\theta_j) + \sin\theta_j = 0, \qquad j = 2,3, \ldots N - 1$$

$$\frac{1}{a^2}(\theta_2 - 2\theta_1) + \sin\theta_1 = 0, \qquad j = 1$$

$$\frac{1}{a^2}(\theta_{N-1} - 2\theta_N) + \sin\theta_N = 0, \qquad j = N$$

A system of coupled nonlinear algebraic equations

Basis-set (using Fourier):

$$\frac{d^2\theta}{d\tau^2} + \sin\theta = 0, \qquad \theta(0) = 0, \theta(\tau_0) = 0$$

$$\theta(\tau) = \sum_n c_n e^{in\omega_0\tau}, \omega_0 = 2\pi/\tau_0$$

$$-(n\omega_0)^2 c_n + \frac{1}{\tau_0}\int_0^{\tau_0} e^{-in\omega_0\tau} \sin\left[\sum_k c_k e^{ik\omega_0\tau}\right] d\tau = 0,$$

A (much more complicated!) system of coupled nonlinear integral equations

Summary:

- Grid-point discretization is much easier to implement for ODEs and PDEs with variable coefficients.

- Basis-set discretisation is only useful if the convergence is good (smaller size matrix problems compared to grid-point).

  Need to be smart with the choice of the basis! Fourier series is not necessarily the best option.

- Basis-set is generally not a good idea for solving nonlinear problems

  BUT we will re-consider this problem later…