993SM - Laboratory of Computational Physics XII week December 10, 2025

Maria Peressi
Università degli Studi di Trieste - Dipartimento di Fisica
Sede di Miramare (Strada Costiera 11, Trieste)
e-mail: peressi@units.it

The Numerov's method for the 1D Schroedinger equation

codes & notes from:

prof. Paolo Giannozzi (UniUD)

"Numerical methods in Quantum Mechanics"

https://www.fisica.uniud.it/~giannozz/Corsi/MQ/LectureNotes/mq.pdf

https://www.fisica.uniud.it/~giannozz/Corsi/MQ/Software/F90/harmonic0.f90

https://www.fisica.uniud.it/~giannozz/Corsi/MQ/Software/F90/harmonic1.f90

Note:
we choose a problem that can be solved exactly
(analytically)
to check the reliability of the code
and the possible problems

the harmonic oscillator

harmonic oscillator:
classical

Force $F = -Kx$

$$m\frac{d^2x}{dt^2} = -Kx$$

Potential $V(x) = V(-x) = \frac{1}{2}Kx^2$

A solution $x(t) = x_0 \sin(\omega t)$ with $\omega = \sqrt{\frac{K}{m}}$

probability $\rho(x)dx$ to find the mass between x and $x+dx$:

$$\rho(x)dx \propto \frac{dx}{v(x)}$$

Since $v(t) = x_0\omega\cos(\omega t) = \omega\sqrt{x_0^2 - x_0^2\sin^2(\omega t)}$, we have

$$\rho(x) \propto \frac{1}{\sqrt{x_0^2 - x^2}}$$

for $|x| < x_0$; 0 elsewhere

harmonic oscillator:
1D Schroedinger eq.

In standard notation:

vpot

$$\frac{d^2\psi}{dx^2} = -\frac{2m}{\hbar^2}\left(E - \frac{1}{2}Kx^2\right)\psi(x) \qquad \text{with} \qquad \omega = \sqrt{\frac{K}{m}}$$

Note the symmetry of the potential: $V(-x) = V(x)$

Defining:

$$\xi = \left(\frac{m\omega}{\hbar}\right)^{1/2} x = \left(\frac{mK}{\hbar^2}\right)^{1/4} x = \frac{x}{\lambda} \quad \text{and} \quad \varepsilon = \frac{E}{\hbar\omega}$$

we rewrite the eq. in adimensional units:

$$\frac{d^2\psi}{d\xi^2} = -2\left(\varepsilon - \frac{\xi^2}{2}\right)\psi(\xi)$$

vpot in harmonic0.f90

harmonic oscillator:
1D Schroedinger eq.

Exact solution (analytical):

$$\psi_n(\xi) = H_n(\xi)\mathrm{e}^{-\xi^2/2}$$

odd or even functions

n nodes and the same parity as n

Hermite polynomials. $H_n(\xi)$

The lowest-order Hermite polynomials are

$$H_0(\xi) = 1, \quad H_1(\xi) = 2\xi, \quad H_2(\xi) = 4\xi^2 - 2, \quad H_3(\xi) = 8\xi^3 - 12\xi.$$

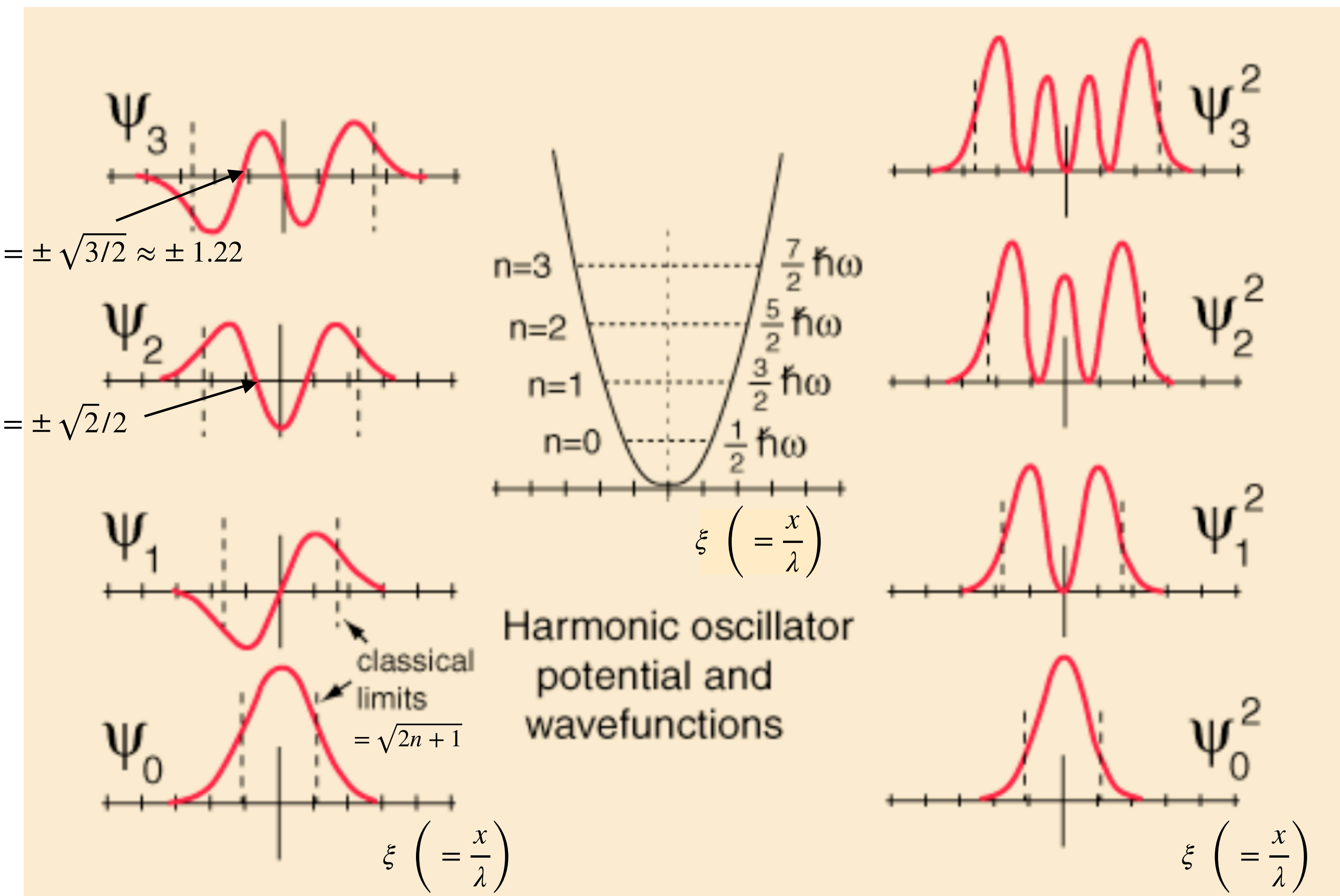
corresponding to discretized energies:

$$\varepsilon = n + \frac{1}{2} \Rightarrow E_n = \left(n + \frac{1}{2}\right)\hbar\omega \quad n = 0, 1, 2, \ldots$$

n is a non-negative integer

harmonic oscillator:
1D Schroedinger eq.

Exact solution (plots):



$\mathscr{H}_3(\xi_0) = 0 \rightarrow \xi_0 = \pm\sqrt{3/2} \approx \pm 1.22$

$\mathscr{H}_2(\xi_0) = 0 \rightarrow \xi_0 = \pm\sqrt{2}/2$

The Numerov's method

To solve:

(g(x), s(x) given)

$$\frac{d^2y}{dx^2} = -g(x)y(x) + s(x)$$

idea:

- consider a box [$-x_{max}$: x_{max}] out of which we expect y(x) to be negligible;
- discretize [$-x_{max}$: x_{max}] into small N intervals Δx (x_n are the corresponding points);
- around each x_n do a Taylor expansion of y(x), g(x), s(x) (y_n=y(x_n) and similar; backwards and forwards, for y_{n-1} and y_{n+1})
- a few manipulations follow…

(details in the notes by prof. Giannozzi)

$$\begin{aligned} y_{n+1}\left[1+g_{n+1}\frac{(\Delta x)^2}{12}\right] &= 2y_n\left[1-5g_n\frac{(\Delta x)^2}{12}\right] - y_{n-1}\left[1+g_{n-1}\frac{(\Delta x)^2}{12}\right] \\ &+(s_{n+1}+10s_n+s_{n-1})\frac{(\Delta x)^2}{12} + O[(\Delta x)^6] \end{aligned}$$

allows to obtain y_{n+1} starting from y_n and y_{n-1}, and recursively go on…

1D Schroedinger equation:
a form suitable for Numerov's method

The Scroedinger eq.:

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

has the form:

$$\frac{d^2y}{dx^2} = -g(x)y(x) + s(x)$$

with:

$$g(x) = (2m/\hbar^2)[E - V(x)] \text{ and } s(x) = 0$$

1D Schroedinger equation: harmonic oscillator - 1

In adimensional units:

$$\frac{d^2\psi}{d\xi^2} = -2\left(\varepsilon - \frac{\xi^2}{2}\right)\psi(\xi)$$

has the form:

$$\boxed{\frac{d^2y}{dx^2} = -g(x)y(x) + s(x)}$$

with:

$$g(x) = 2\left(\epsilon - \frac{x^2}{2}\right) \quad \text{and} \quad s(x) = 0$$

1D Schroedinger equation: harmonic oscillator - 2

Since $s(x) = 0$, the Numerov's formula reduces to :

$$y_{n+1}\left[1 + g_{n+1}\frac{(\Delta x)^2}{12}\right] = 2y_n\left[1 - 5g_n\frac{(\Delta x)^2}{12}\right] - y_{n-1}\left[1 + g_{n-1}\frac{(\Delta x)^2}{12}\right] + O[(\Delta x)^6]$$

Defining: $f_n \equiv 1 + g_n\dfrac{(\Delta x)^2}{12}$

we rewrite Numerov's formula as

$$y_{n+1} = \frac{(12 - 10f_n)y_n - f_{n-1}y_{n-1}}{f_{n+1}}$$

that allows to obtain y_{n+1} starting from y_n and y_{n-1}, and recursively the function in the entire box.

The value of the energy is now hidden into g_n and f_n.

$$g_n = 2\left(\epsilon - \frac{x_n^2}{2}\right)$$

1D Schroedinger equation: harmonic oscillator - 3

$$y_{n+1} = \frac{(12 - 10 f_n) y_n - f_{n-1} y_{n-1}}{f_{n+1}} \qquad (*)$$

The symmetry of the potential and the parity of the (still unknown) solutions allows to simplify the choice of the starting points

Number of nodes:

odd *(dispari)* hence $y(0) = 0$ — choose $y_0 = 0$ and whatever y_1 you want

even *(pari)* hence $y(-x) = y(x)$ — choose whatever y_0 (finite) you want;

y_1 is determined by Numerov's formula :

since $f_{-1} = f_1$ by symmetry, and $y_{-1} = y_1$; put into $(*)$ and obtain :

$$y_1 = \frac{(12 - 10 f_0) y_0}{2 f_1}$$

harmonic0

The code prompts for some input data:

- the limit $x_{\max}$ for integration (typical values: $5 \div 10$);
- the number N of grid points (typical values range from hundreds to a few thousand); note that the grid point index actually runs from 0 to N, so that $\Delta x = x_{\max}/N$;
- the name of the file where output data is written;
- the required number of nodes (the code will stop if n is negative).

Finally the code prompts for a trial energy. You should answer 0 (*) in order to search for an eigenvalue with n nodes. The code will start iterating on the energy

(*) It is however possible to specify an energy to force the code to perform an integration at fixed energy useful for testing purposes

new value of the number of nodes: answer -1 to stop

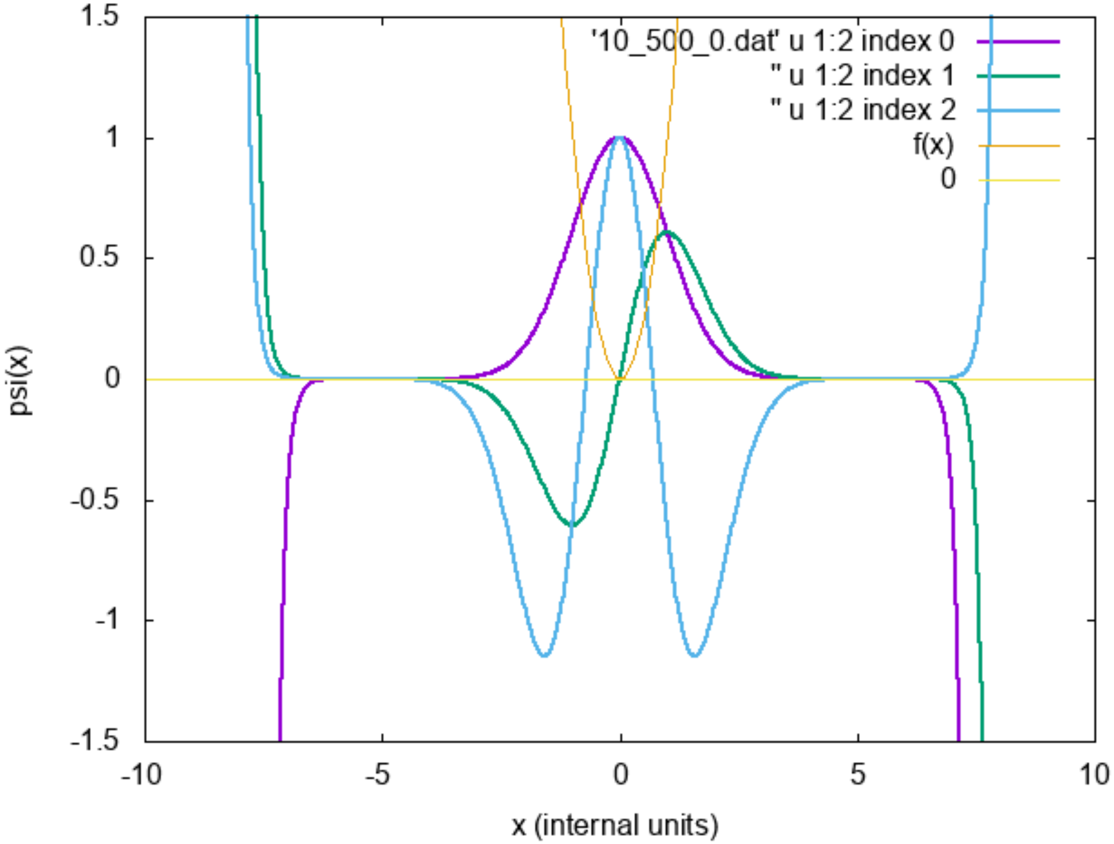
harmonic0

```
Max value for x (typical value: 10) > 10
Number of grid points (typically a few hundreds) > 500
output file name > 10_500.dat
nodes (type -1 to stop) > 0
Trial energy (0=search with bisection) > 0
           1    25.000000000000000              13           0
           2    12.500000000000000               7           0
           3    6.2500000000000000               3           0
           4    3.1250000000000000               2           0
```

Output file contains: $x,\ \psi(x),\ |\psi(x)|^2,\ \rho_{\rm cl}(x),\ V(x)$

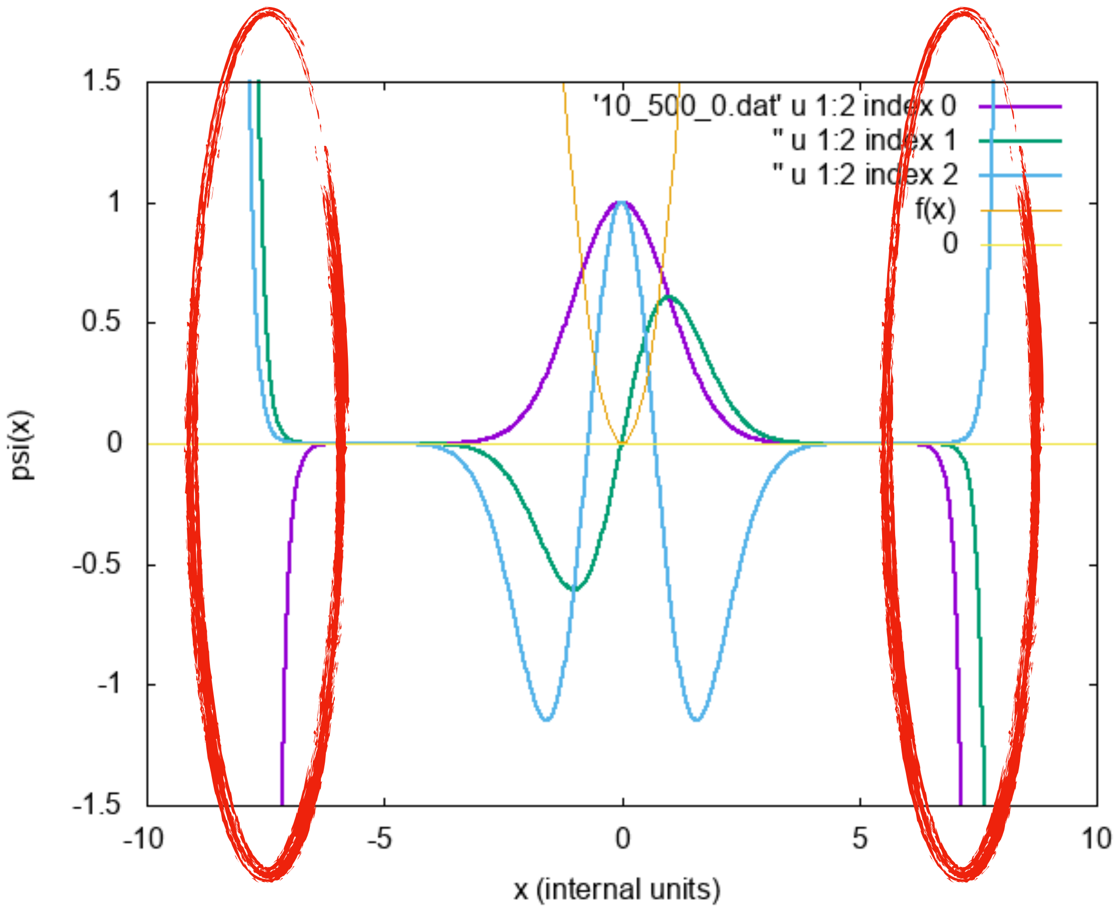
```
#    x          y(x)            y(x)^2         classical p(x)      V
-10.000 -0.41559380E+11  0.17271820E+22  0.00000000E+00   50.000000
 -9.980 -0.34101628E+11  0.11629210E+22  0.00000000E+00   49.800200
 -9.960 -0.27993466E+11  0.78363416E+21  0.00000000E+00   49.600800
 -9.940 -0.22988666E+11  0.52847878E+21  0.00000000E+00   49.401800
 -9.920 -0.18886279E+11  0.35669155E+21  0.00000000E+00   49.203200
```

Results



harmonic0

Results



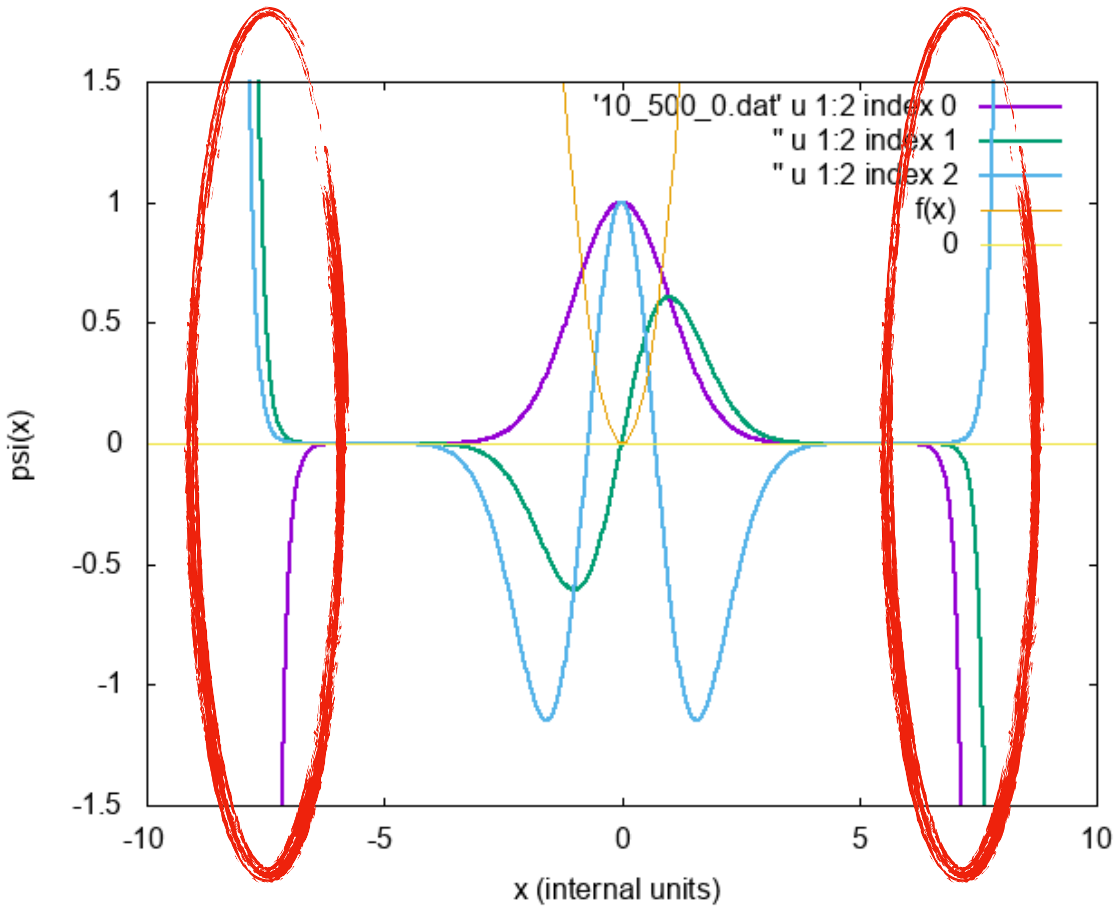
???

divergences!

… but can be hidden!

harmonic0

Results



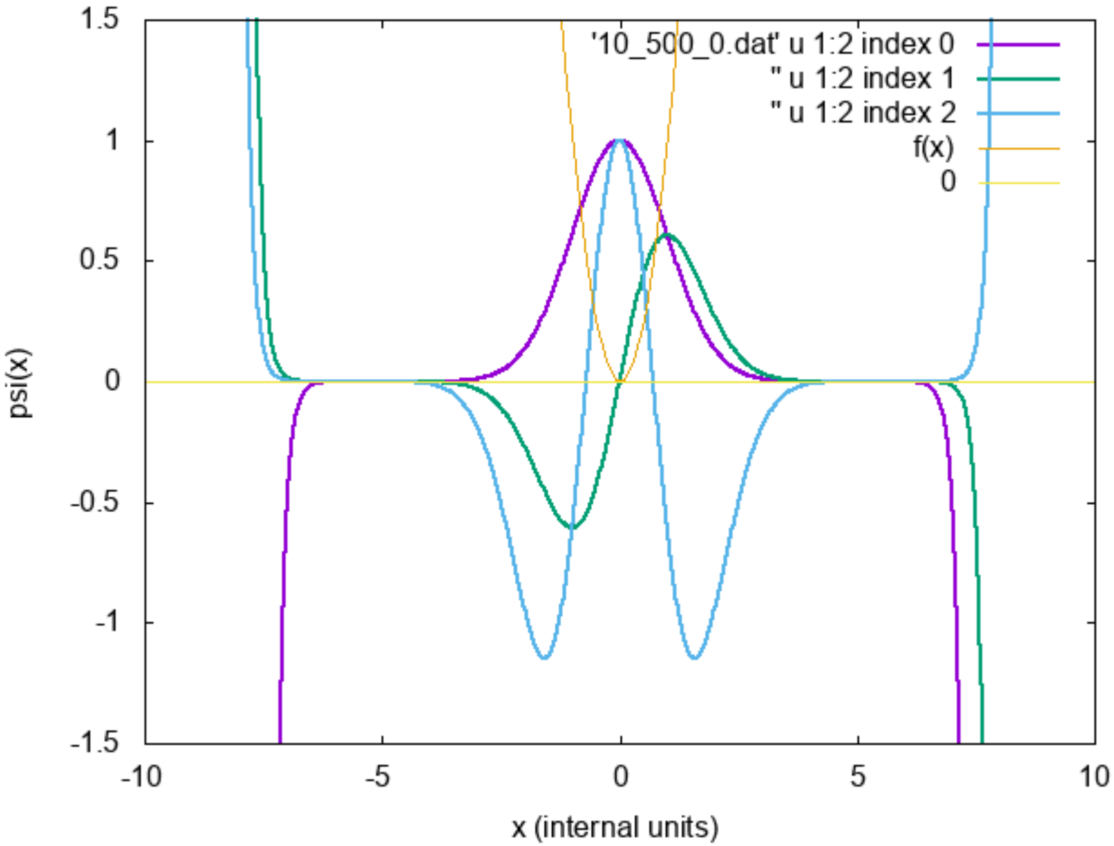
harmonic0

???

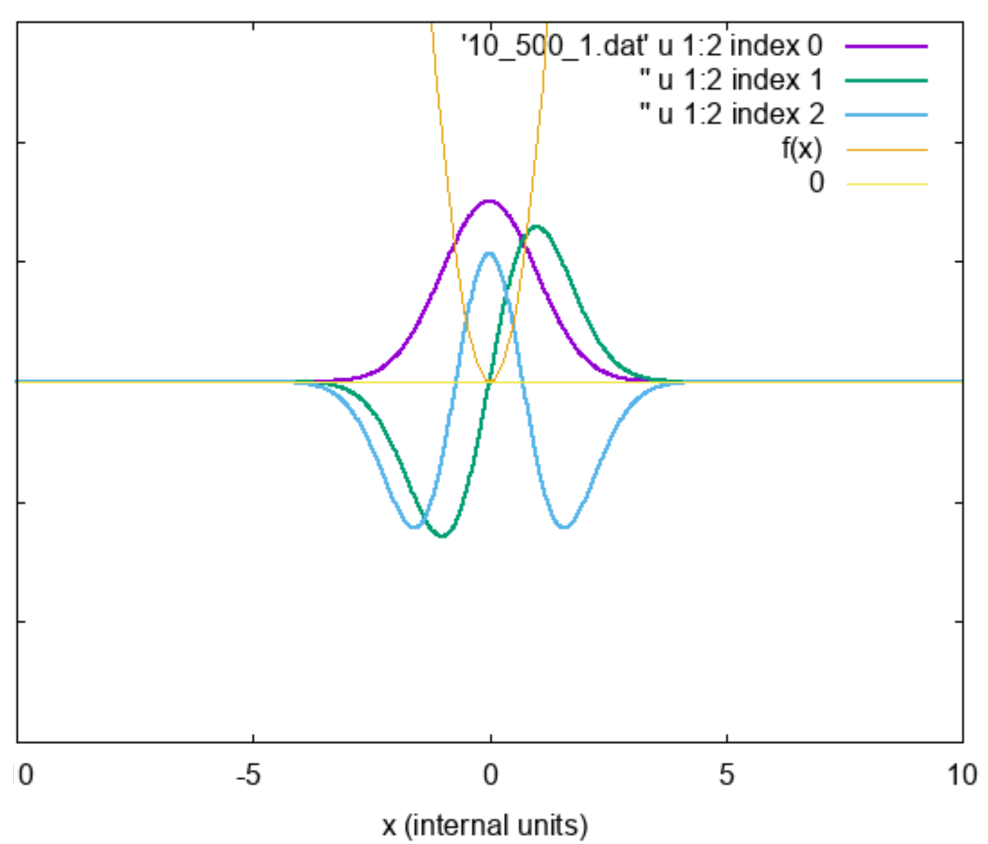
divergences!
… but can be hidden!

what's next?
integration
forward (from x=0 to x_max) and
backward (from x_max to 0)
=> harmonic1

Results



harmonic0



harmonic1

sneaking a look at the Numerov's method

Find a solution ψ with a given number `n` of nodes and energy `En`

We do not know `En`, but we start from an initial energy `E`, average of `[Elow, Eup]` that we know for sure to contain En

Start integrating ψ from `x=0` towards positive values of `x` (forward) (*)

During integration, count for the number `icl` of changes of sign of ψ:
if `icl>n` => `E` is too high => choose `[Elow, E]`
if `icl<n` => `E` is too low => choose `[E, Eup]`

(*) From the initial point it is obviously possible to integrate by moving both in the direction of the positive x and in that of the negative x, and in the presence of symmetry with respect to an inversion point it will be sufficient to integrate in only one direction.

within the Numerov's method: the "shooting" step

Used to count the number of nodes while "building" the wavefunction:

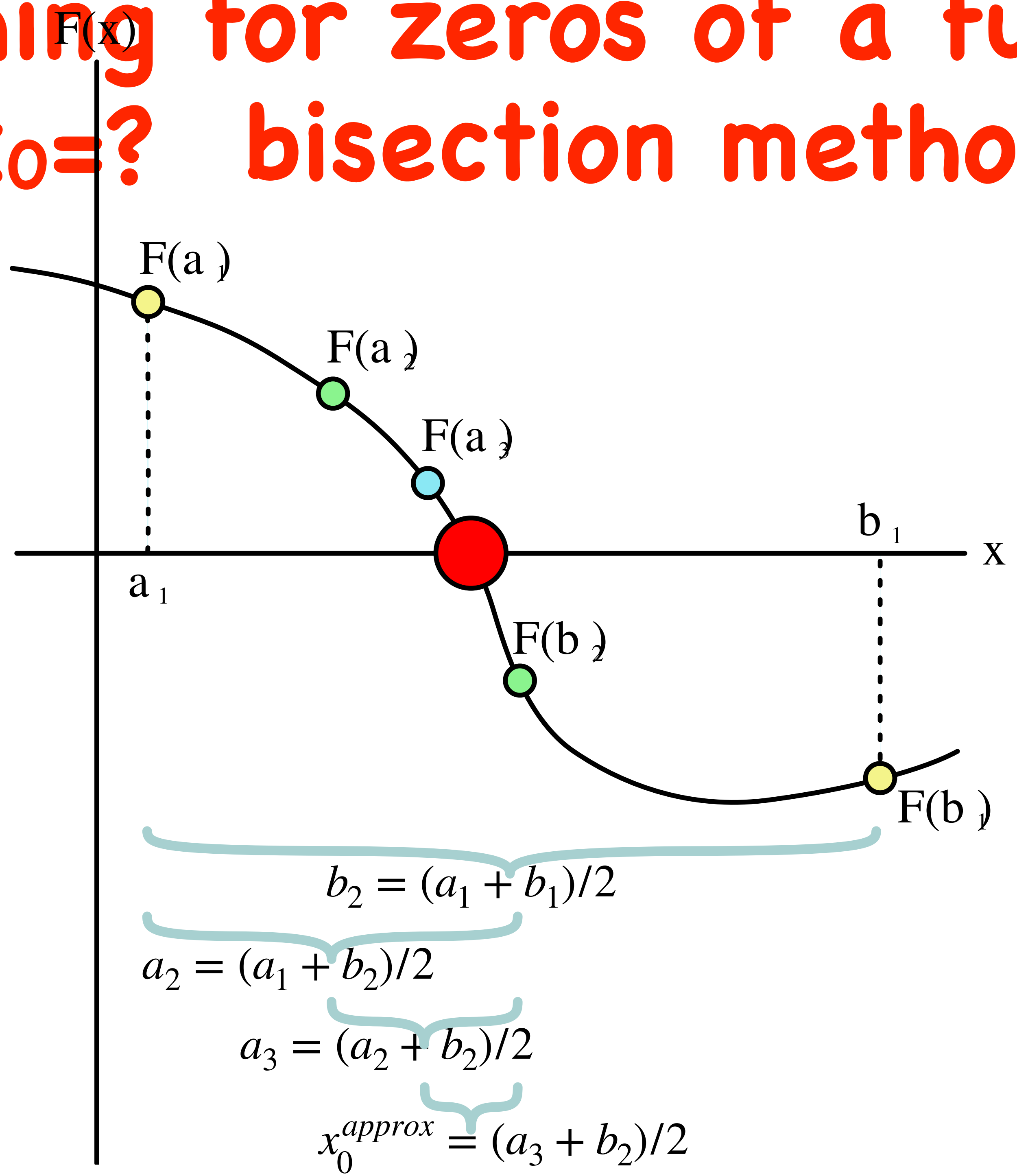
```
if ( y(i) /= sign(y(i),y(i+1)) ) ncross=ncross+1
```

`SIGN(A,B)` returns the value of *A* with the sign of *B*

Used to count the number of change of sign of g(x), determining the classical inversion point.
Note: this check is done before redefining f in this way:
f = 1.0_dp - f
therefore the check is actually on g(x)

```
f(i)=ddx12*(2.0_dp*(vpot(i)-e))
if ( f(i) /= sign(f(i),f(i-1)) ) icl=i
```

searching for zeros of a function
x_0=? bisection method



searching for zeros of a function x_0=? bisection method

1. determine an interval $[x_L,x_U]$ at whose extremes the function $y(x)$ has discordant signs (to be sure that it contains x_0)
2. calculate the midpoint of the interval $[x_L,x_U]$: $x_M = (x_L + x_U)/2$ and evaluate $y(x_M)$;
3. If $y(x_M)=0$ then $x_M = x_0$ and the search ends.
4. Otherwise, take as the new interval the one at whose extremities the function has discordant signs (depending on the case it will be necessary to redefine $x_L=x_M$ or $x_U=x_M$): it contains x_0
5. iterate points 2 - 4 until:

a) the uncertainty on the location of x_0 decreases below a pre-established an absolute threshold ($|x_U-x_L| < \varepsilon$), or a relative threshold ($|x_U-x_L| < \varepsilon\ |x_L|$ or $< \varepsilon\ |x_U|$, where x_0 is approximated by x_L or x_U); or

b) $|y(x_M)| < \varepsilon'$; or

c) a maximum number of iterations is exceeded.

searching for zeros of a function x_0=? bisection method

1. determine an interval $[x_L, x_U]$ at whose extremes the function y(x) has discordant signs (to be sure that it contains x_0)

Implementation:

```
y(xL)* y(xU) < 0
```

`y(xL)/= sign(y(xL),y(xU))` or `y(xU)/= sign(y(xU),y(xL))`

In harmonic0.f90:

```
if ( y(i) /= sign(y(i),y(i+1)) ) ncross=ncross+1
```

(`ncross` means crossing with the x axis, i.e., zeros)

searching for zeros of a function x_0=? bisection method

4. take as the new interval the one at whose extremities the function has discordant signs (depending on the case it will be necessary to redefine $x_L = x_M$ or $x_U = x_M$): it contains x_0

Implementation:

```
if (y(xL)*y(xM) < 0) then
    xU = xM
else if (y(xU)*y(xM) < 0) then
    xL = xM
end if
```

searching for zeros of a function $x_0 = ?$ bisection method

5. iterate points 2 - 4 until:

a) the uncertainty on the location of x_0 decreases below a pre-established an absolute threshold ($|x_U - x_L| < \varepsilon$), or a relative threshold ($|x_U - x_L| < \varepsilon\, |x_L|$ or $< \varepsilon\, |x_U|$, where x_0 is approximated by x_L or x_U)

Implementation - which is the best criterion?

- use an absolute threshold: possible problems (for roundoff errors) if x_0 is large
- use a relative threshold: possible problems if x_0 is small
- other possible problems if $y(x)$ is too flat close to x_0

harmonic1

uses Forward and backward integration