11.2.7 Finding Eigenvalues by the Shooting Method

The shooting method is a means to compute eigenvalues of a boundary problem defined by a one-dimensional differential equation and boundary conditions fixed at the end of the interval whereinthe equation is defined.
At first a trial value is inserted for the eigenvalue and the differential equationis solved by numeric integration for an initial problem with initial data fulfilling the boundary condition(s) at one end. The integration goes as far as the other end. In most cases inspection of the solution shows that there,the boundary condition(s) is (are) not fulfilled. But the result gives a hint how to improve the trial value for the eigenvalue. This procedure is repeated till a solution fulfilling all boundary conditions is obtained. The trial value then used is an approximation to an eigenvalue of the boundary value problem.

This method is displayed for the eigenvalue problem of the one-dimensional harmonic oscillator in quantum mechanics. The differential equation expressed in dimensionless variables is :

$$\frac{\partial^2 y}{\partial x^2} + \left(2\,\nu + 1\ - x^2\right) y\ =\ 0$$

ν is the eigenvalue parameter; it is related to energy E by $E = \hbar\omega\,(2\,\nu + 1)/2$. The boundary cond-tions are:

$$\mathbf{x = \pm\,\infty:\ \ y = 0.}$$

Analytic solution leads to $E/\hbar\omega\ =\ 1/2,\ 3/2,\ ...,$ i.e. $\nu = 0,\ 1,\ ...$; see § 11.1.1 This is now shown by the shooting method. In place of x = ± ∞ a value of sufficient magnitude is chosen; experience shows that for low ν x = 5 suffices; the value of the first derivative must be small, the specific value is uncritical. The end value xe is determined empirically such that the resulting curve fits into the picture.

```
Clear[x, y, sh, nu, xe]
```

11.2.7.1 Showing the shooting method in detail for n = 0

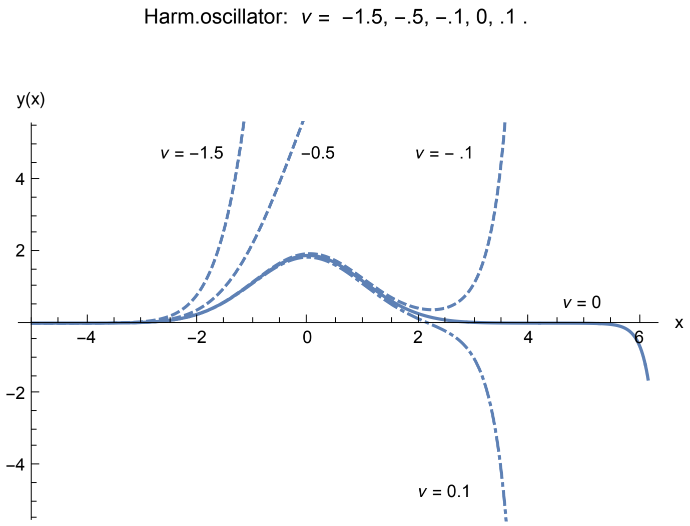
```
sh[nu_,xe_] :=
NDSolve[ {y''[x]  + (2 nu + 1 - x^2) y[x] == 0,
    y[-5] == 4 10^-6, y'[-5] == 5 10^-5}, y[x], {x,-5,xe}]

SetOptions[Plot, PlotStyle -> Thickness[.005]];
dd = PlotStyle -> Dashing[{.01}];
dt = PlotStyle -> Dashing[{.02,.01,.0025,.01}];

so = Table[0, {5}]; pp = so;

so⟦1⟧ = sh[-1.5`, xe = -0.5`];
pp⟦1⟧ = Plot[Evaluate[y[x] /.so⟦1⟧], {x, -5, xe}, Evaluate[dd]];
so⟦2⟧ = sh[-0.5`, xe = 1];
pp⟦2⟧ = Plot[Evaluate[y[x] /.so⟦2⟧], {x, -5, xe}, Evaluate[dd]];
so⟦3⟧ = sh[-0.01`, xe = 6];
pp⟦3⟧ = Plot[Evaluate[y[x] /.so⟦3⟧], {x, -5, xe}, Evaluate[dd]]; so⟦4⟧ = sh[0, xe = 7];
pp⟦4⟧ = Plot[Evaluate[y[x] /.so⟦4⟧], {x, -5, xe}]; so⟦5⟧ = sh[0.01`, xe = 6];
pp⟦5⟧ = Plot[Evaluate[y[x] /.so⟦5⟧], {x, -5, xe}, Evaluate[dt]];
```

```
Show[pp⟦1⟧, pp⟦2⟧, pp⟦3⟧, pp⟦4⟧, pp⟦5⟧, PlotRange → {-5, 5}, AxesLabel → {"x", "y(x)"},
 PlotLabel → "Harm.oscillator:  ν =  -1.5, -.5, -.1, 0, .1 . \n\n",
 PlotRegion → {{0.01`, 0.99`}, {0.01`, 0.99`}},
 Epilog → {Text["ν = -1.5", {-2.1, 4.75}], Text[-0.5, {0.19, 4.75}],
   Text["ν = - .1", {2.47, 4.75}], Text["ν = 0.1", {2.47, -4.75}],
   Text["ν = 0 ", {5, 0.55}]}]
```



The boundary conditions are best approximated by the continuous curve corresponding to ν = 0. Note the different behaviour of the solutions belonging to ν < 0 (dashed curves) or ν > 0 (dashed-dotted curves).
Since the curve for   ν = 0 has no node in the interior of the interval, the eigenvalue must be the lowest one.

11.2.7.2 Automating the routine for finding the eigenvalues

Now the method is automated so that a routine searches for the characteristic values. A function is defined to get the end value y(xe) as a function of ν =nu. This function is used in **FindRoot[]**. In order to avoid troubles with FindRoot[] one must either

1) apply the operator **Hold[]** to the definition of tr[n_u]. This holds the evaluatin till te[nu] is used in **FindRoot[]**.
   This hold is released by the command **Releas**e. or
2) restrict the arguments of the function te[] to numeric values as done at the end of this subsubsection.

```
xe = 5;

tet[nu_] := Hold[y[xe] /. NDSolve[ {y''[x]  + (2 nu + 1 - x^2) y[x] == 0,
    y[-5] == 4 10^-6, y'[-5] == 5 10^-5}, y, {x,-5,xe}]]

FindRoot[tet[nnn], {nnn, .8, 1.3}]
tet[nnn /. %] // Release

{nnn → 1.}

{-8.97932 × 10^-9}

FindRoot[tet[nnn], {nnn, 1.8, 2.3}]
tet[nnn /. %] // Release

{nnn → 2.}

{8.34379 × 10^-12}
```

```
FindRoot[tet[nnn], {nnn, 2.8, 3.3}]
tet[nnn /. %] // Release
```

$\{\text{nnn} \to 3.\}$

$\{-8.51405 \times 10^{-12}\}$

11.2.7.3 The variant wherein nu is restricted to numeric values

```
ten[nu_?NumericQ] := y[xe] /. NDSolve[ {y''[x]  + (2 nu + 1 - x^2) y[x] == 0,
    y[-5] == 4 10^-6, y'[-5] == 5 10^-5}, y, {x,-5,xe}]
```

```
FindRoot[ten[nnn], {nnn, .8, 1.3}]
ten[nnn /. %]
```

$\{\text{nnn} \to 1.\}$

$\{-8.97932 \times 10^{-9}\}$

```
FindRoot[ten[nnn], {nnn, 1.8, 2.3}]
ten[nnn /. %]
```

$\{\text{nnn} \to 2.\}$

$\{8.34379 \times 10^{-12}\}$

11.3 Solutions of Partial Differential Equations.

11.3.1 Analytical Solution

Remember that while an ordinary differential equation determines the solution up to some integration constants the solution of a partial differential equation contains arbitray functions. These are denoted as C[i] (i being some integer) and are arbitray except that a suffcient number of derivatives must be continuous.

```
deq = y D[u[x, y], x] + x D[u[x, y], y] == 0
```

$x\, u^{(0,1)}[x, y] + y\, u^{(1,0)}[x, y] == 0$

```
so = DSolve[deq, u[x, y], {x, y}]
```

$$\left\{\left\{u[x, y] \to C[1]\left[\frac{1}{2}\left(-x^2 + y^2\right)\right]\right\}\right\}$$

```
deq = y D[u[x, y], x] + x D[u[x, y], y] == 1
```

$x\, u^{(0,1)}[x, y] + y\, u^{(1,0)}[x, y] == 1$

```
so = DSolve[deq, u[x, y], {x, y}]
```

$$\left\{\left\{u[x, y] \to -\text{Log}\left[x + \sqrt{y^2}\right] + C[1]\left[\frac{1}{2}\left(-x^2 + y^2\right)\right]\right\},\right.$$
$$\left.\left\{u[x, y] \to \text{Log}\left[x + \sqrt{y^2}\right] + C[1]\left[\frac{1}{2}\left(-x^2 + y^2\right)\right]\right\}\right\}$$

```
deq = y D[u[x, y], x] - x D[u[x, y], y] == 0
```

$-x\, u^{(0,1)}[x, y] + y\, u^{(1,0)}[x, y] == 0$

```
so = DSolve[deq, u[x, y], {x, y}]
```

$$\left\{\left\{u[x, y] \to C[1]\left[\frac{1}{2}\left(x^2 + y^2\right)\right]\right\}\right\}$$

```
deq = x D[u[x, y, z], x] + y D[u[x, y, z], y] +  z D[u[x, y, z], z] == 0
```

$$z\,u^{(0,0,1)}[x,y,z]+y\,u^{(0,1,0)}[x,y,z]+x\,u^{(1,0,0)}[x,y,z]==0$$

```
so = DSolve[deq, u[x, y, z], {x, y, z}]
```

$$\left\{\left\{u[x,y,z]\to C[1]\left[\frac{y}{x},\frac{z}{x}\right]\right\}\right\}$$

```
deq =  D[u[x, y], {x, 2}] + D[u[x, y], {y, 2}]  == 0
```

$$u^{(0,2)}[x,y]+u^{(2,0)}[x,y]==0$$

```
so = DSolve[deq, u[x, y], {x, y}]
```

$$\{\{u[x,y]\to C[1][i\,x+y]+C[2][-i\,x+y]\}\}$$

```
deq =  D[u[x, t], {x, 2}] - 1 / c^2 D[u[x, t], {t, 2}]  == 0
```

$$-\frac{u^{(0,2)}[x,t]}{c^2}+u^{(2,0)}[x,t]==0$$

```
so = DSolve[deq, u[x, t], {x, t}]
```

$$\left\{\left\{u[x,t]\to C[1]\left[t-\frac{x}{\sqrt{c^2}}\right]+C[2]\left[t+\frac{x}{\sqrt{c^2}}\right]\right\}\right\}$$

```
PowerExpand[so] // Flatten
```

$$\left\{u[x,t]\to C[1]\left[t-\frac{x}{c}\right]+C[2]\left[t+\frac{x}{c}\right]\right\}$$

```
deq =  D[u[r, ϕ], {r, 2}] + 1 / r D[u[r, ϕ], r] + 1 / r^2 D[u[r, ϕ], {ϕ, 2}] == 0
```

$$\frac{u^{(0,2)}[r,\phi]}{r^2}+\frac{u^{(1,0)}[r,\phi]}{r}+u^{(2,0)}[r,\phi]==0$$

```
so = DSolve[deq, u[r, ϕ], {r, ϕ}]
```

$$\text{DSolve}\left[\frac{u^{(0,2)}[r,\phi]}{r^2}+\frac{u^{(1,0)}[r,\phi]}{r}+u^{(2,0)}[r,\phi]==0,\ u[r,\phi],\ \{r,\phi\}\right]$$

11.3.2 Numerical Solution of Simple Partial Differential Equations

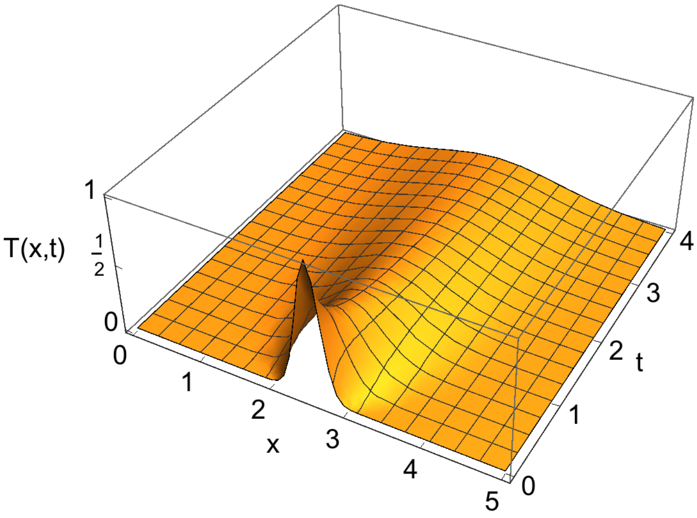
This solves the heat equation in one space dimension with the boundaryconditions $x_{0,1}$ = 0, 5: T(x_i,t) = 0.

At the beginning a temperature peak is present around x= 2.5.

```
sol =
 NDSolve[{ D[T[x, t], t] == D[T[x, t], x, x] / 10, T[x, 0] == Exp[- (x - 2.5) ^2 / 0.05 ],
    T[0, t] == 0, T[5, t] == 0}, T, {x, 0, 5}, {t, 0, 4}] // Flatten
```

```
{T → InterpolatingFunction[ ⊞ Domain: {{0., 5.}, {0., 4.}}
                              Output: scalar ]}
```

```
Plot3D[Evaluate[T[x, t] /. sol], {x, 0, 5},
 {t, 0, 4}, AxesLabel → {"x", "t", "T(x,t)   "},
 Ticks → {Range[0, 5], Range[0, 4], Range[0, 2] / 2},
  PlotRange → All, BaseStyle → FontSize → 14]
```



11.3.3 Numerical Solution of Partial Differential Equations by Difference or Finite Elements Methods

Numerical Solutions of Partial Differential Equations may be found by difference method or finite elements.

For the first method, the partial derivatives are replaced by differences; in this way the differential equations are approximated by large linear systems of equations for the field values at the nodes of the mesh. In the book of

Victor G. Ganzha and Evgenii V. Vorozhtsov: Numerical Solutions for Partial Differential Equations: Problem Solving Using *Mathematica*,

this is shown and programs are given.

Prem K. Kythe, Pratap Puri, Michael R. Schaferkotter, Hiles K. Jones, Michael R. Schaeferkotter:
Partial Differential Equations and Boundary Value Problems with Mathematica
CRC Press; 2 edition, 2002

11.4 Exercises

11.1 A charged particle moves in static, homogeneous, crossed electric and magnetic fields and experiences a linear frictional force.

$$d^2\mathbf{r}/dt^2 = q\,\mathbf{E} + q\,(d\mathbf{r}/dt \times \mathbf{B}) - m\,a\ d\mathbf{r}/dt$$

$$\mathbf{E} = (0, E0, 0),\ \ \mathbf{B} = (0,0,B0),\ \ a = \text{const.}$$

Initial data are:

$\mathbf{r}(0) = (0,0,0)$, $d\mathbf{r}/dt(0) = (vx, vy, vz)$.

Solve the system of differential equations analytically (if possible) and numerically for suitable values of the parameters and initial conditions.

11.2 Same problem as 1., but the frictional force is quadratic, i.e. it is

$$- m\,a\ d\mathbf{r}/dt\ |\ d\mathbf{r}/dt\ |.$$

Compute the trajectory by numeric solution for suitable values of the parameters and prepare a drawing.

11.3. A particle is projected from the ground with a given initial velocity oblique to the field of gravity. Compute and draw the trajectory.
Indicate equal time intervalls by dots on the trajectory.
Do the same for an additional linear or quadratic frictional force.

11.4. A 1-dimensional linear force $F = -x$ acts on a particle as long as $|x| < 1$; the force is zero for $|x| > 1$.
The particle starts with a given initial speed at $x = 0$. Compute and draw the position and the velocity as function of time and draw the phase space diagram for two initial velocities such that the particle is bound and for two initial velocities belonging to unbound motion.

11.5. Compute and draw the Poincare Map for the x,x'-plane of Sect. 11.2.6

11.6. Treat the following differential equation: $y'' + 2\,x\,y' + 2\,y = 0;$
11.6.1 Solve the differential equation analytically.
11.6.2 Solve analytically the boundary values problem: $y(0) = 0,\ y(2) = 1.$
11.6.3 Solve numerically the initial value problem: $y(0) = 1,\ y'(0) = 0;$ plot the solution.

11.7 Verify that $y[x] = \text{Exp}[x]$ is a solution of the following differential equation :

$$y[x] - x\,y'[x] + (-3 + 2\,x)\,y''[x] + (2 - x)\,y^{(3)}[x] = 0.$$

11.8 Compute the eigenvalue $v = 1$ for the quantenmechanical Harmonic Oscillater by the shooting method as it is done for $v = 0$ in subsubsection 11.2.7.1. Plot the curves for v around 1. Show the curvews of the shooting method in detail for $v = 0.8$, 0.9, 1.0 and 1.1 .