Numerical Simulation of Dynamic Systems: Hw7 - Solution

Prof. Dr. François E. Cellier
Department of Computer Science
ETH Zurich

April 23, 2013

[H6.1a] Interpolation of Measurement Data

Given a set of measurement data of surface air temperature:

t [hours]	u °C
0	6
6	16
12	28
18	21
24	18
36	34
48	18
60	25
66	15
72	4

where t denotes the time measured in hours, and u denotes the surface air temperature at that time measured in degrees Centigrade.

[H6.1a] Interpolation of Measurement Data

Given a set of measurement data of surface air temperature:

t [hours]	u °C
0	6
6	16
12	28
18	21
24	18
36	34
48	18
60	25
66	15
72	4

where t denotes the time measured in hours, and u denotes the surface air temperature at that time measured in degrees Centigrade.

We shall need intermediate values, and therefore, we require an *interpolation routine*.

[H6.1a] Interpolation of Measurement Data II

Matlab offers a cubic spline routine for this purpose.

[H6.1a] Interpolation of Measurement Data II

Matlab offers a cubic spline routine for this purpose.

Cubic splines place cubic polynomials in each segment, i.e., in the range of values between two neighboring measurement data points, such that:

1. the values of the interpolation polynomial are correct at the two measurement data points, and
2. the first derivatives at these measurement data points coincide with the first derivatives of the interpolation polynomials of the neighboring segment.

[H6.1a] Interpolation of Measurement Data II

Matlab offers a cubic spline routine for this purpose.

Cubic splines place cubic polynomials in each segment, i.e., in the range of values between two neighboring measurement data points, such that:

1. the values of the interpolation polynomial are correct at the two measurement data points, and
2. the first derivatives at these measurement data points coincide with the first derivatives of the interpolation polynomials of the neighboring segment.

Make use of the spline routine to obtain interpolated data points at all multiples of six hours, such that the new "measurement" data are equidistantly spaced over the range of values $t \in \{0\}, \{6\}, \ldots, \{72\}$ *hours*.

[H6.1a] Interpolation of Measurement Data III

We now wish to compare two different interpolation routines:

1. cubic spline interpolation
2. 3^{rd}-order Nordsieck interpolation

To this end, we need to develop a Matlab function nordsieck_intpol that has the same parameters as the spline function.

[H6.1a] Interpolation of Measurement Data III

We now wish to compare two different interpolation routines:

1. cubic spline interpolation
2. 3^{rd}-order Nordsieck interpolation

To this end, we need to develop a Matlab function nordsieck_intpol that has the same parameters as the spline function.

We start out with the formula:

$$\begin{pmatrix} x_k \\ h \cdot \dot{x}_k \\ \frac{h^2}{2} \cdot \ddot{x}_k \\ \frac{h^3}{6} \cdot x_k^{(\mathrm{iii})} \end{pmatrix} = \frac{1}{6} \cdot \begin{pmatrix} 6 & 0 & 0 & 0 \\ 11 & -18 & 9 & -2 \\ 6 & -15 & 12 & -3 \\ 1 & -3 & 3 & -1 \end{pmatrix} \cdot \begin{pmatrix} x_k \\ x_{k-1} \\ x_{k-2} \\ x_{k-3} \end{pmatrix} \tag{1}$$

and move the grid by the minimal amount necessary to make one of the grid points coincide with the desired interpolation point.

[H6.1a] Interpolation of Measurement Data IV

Calculate the interpolated temperature twice for every full hour, once using spline interpolation and once using Nordsieck interpolation.

[H6.1a] Interpolation of Measurement Data IV

Calculate the interpolated temperature twice for every full hour, once using spline interpolation and once using Nordsieck interpolation.

Plot the two curves on top of each other.

[H6.1a] Interpolation of Measurement Data IV

Calculate the interpolated temperature twice for every full hour, once using spline interpolation and once using Nordsieck interpolation.

Plot the two curves on top of each other.

What do you conclude?

[H6.1a] Interpolation of Measurement Data V

I chose to normalize the support values in the same way as I did for the Newton-Gregory polynomials, i.e., $x = 0$ is the leftmost normalized measurement point, $x = 1$ is the next normalized measurement point to the right, etc.

```
[n, m] = size(xvec);
nm = n * m;
x0 = xvec(1);
dx = xvec(2) - xvec(1);
xvec = 0 : nm - 1;
xvec_new = (xvec_new - x0*ones(size(xvec_new)))/dx;
[n1, m1] = size(xvec_new);
nm1 = n1 * m1;
yvec_new = zeros(n1, m1);
```

Here, $xvec$ is the vector of normalized support values, and $xvec_{new}$ is the vector of normalized values, at which we wish to interpolate.

[H6.1a] Interpolation of Measurement Data VI

Since we wish to use the 3^{rd}-order Nordsieck vector, we always need four neighboring support values for the interpolation. We need to determine, which values to use.

```
x = xvec_new(i);
ix = floor(x);
if ix == 0,
    indx = 4 : -1 : 1;
    c = 1;
elseif ix >= nm - 2,
    indx = nm : -1 : nm - 3;
    c = 3;
else
    indx = ix + 3 : -1 : ix;
    c = 2;
end
```

The *indx* vector computes the indices of the four neighboring support values. We need to treat the leftmost and rightmost segments differently. $c = 1$ denotes the leftmost segment, $c = 3$ marks the rightmost segment, and $c = 2$ handles all of the interior segments in between.

[H6.1a] Interpolation of Measurement Data VII

We need to determine, by how much the support values must be moved, i.e., we must compute the value of the normalized dx_{new}. If the value where we wish to interpolate is in the left half of a segment, we shall shorten dx, otherwise we shall lengthen dx. This rule holds everywhere except in the rightmost segment, where we must always shorten dx.

```
d = x - ix;
if c == 1,
    if d < 0.5,
        dxn = (3 - d)/3;
    else
        dxn = (3 - d)/2;
    end
elseif c == 2,
    if d < 0.5,
        dxn = (2 - d)/2;
    else
        dxn = (2 - d);
    end
else
    dxn = 1 - d;
end
```

[H6.1a] Interpolation of Measurement Data VIII

Now, we can move the support values:

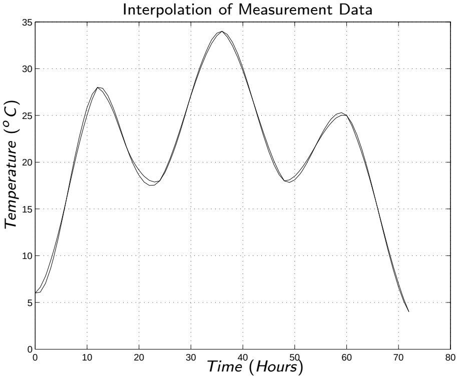
$$
\begin{aligned}
&dxn2 \;=\; dxn * dxn;\\
&dxn3 \;=\; dxn2 * dxn;\\
&H \;= \mathrm{diag}([\;1\;,\;dxn\;,\;dxn2\;,\;dxn3\;]);\\
&v \;=\; yvec(indx);\\
&vnew \;=\; T \backslash H * T * v;
\end{aligned}
$$

[H6.1a] Interpolation of Measurement Data IX

Finally, we need to pick the correct shifted support value:

```
if c == 1,
    if d < 0.5,
        y = vnew(4);
    else
        y = vnew(3);
    end
elseif c == 2,
    if d < 0.5,
        y = vnew(3);
    else
        y = vnew(2);
    end
else
    y = vnew(2);
end
if x == nm - 1,
    y = vnew(1);
end
yvec_new(i) = y;
```

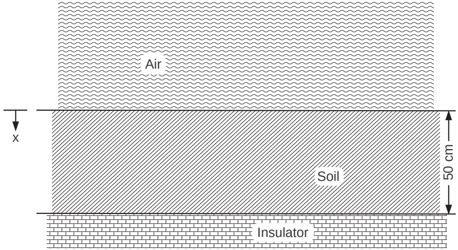
[H6.1a] Interpolation of Measurement Data X



The interpolation worked correctly, but not as smoothly as the cubic spline interpolation.

[H6.1b] Heat Diffusion in the Soil

Agricultural engineers are interested in knowing the temperature distribution in the soil as a function of the surface air temperature. As shown below, we assume that we have a soil layer of 50 cm. Underneath the soil, there is a layer that acts as an ideal heat insulator.



[H6.1b] Heat Diffusion in the Soil II

The heat flow problem can be written as:

$$\frac{\partial u}{\partial t} = \frac{\lambda}{\rho \cdot c} \cdot \frac{\partial^2 u}{\partial x^2}$$

where $\lambda = 0.004 \ \mathrm{cal\ cm^{-1}\ sec^{-1}\ K^{-1}}$ is the specific thermal conductance of soil, $\rho = 1.335 \ \mathrm{g\ cm^{-3}}$ is the density of soil, and $c = 0.2 \ \mathrm{cal\ g^{-1}\ K^{-1}}$ is the specific thermal capacitance of soil.

[H6.1b] Heat Diffusion in the Soil II

The heat flow problem can be written as:

$$\frac{\partial u}{\partial t} = \frac{\lambda}{\rho \cdot c} \cdot \frac{\partial^2 u}{\partial x^2}$$

where $\lambda = 0.004 \text{ cal cm}^{-1} \text{ sec}^{-1} \text{ K}^{-1}$ is the specific thermal conductance of soil, $\rho = 1.335 \text{ g cm}^{-3}$ is the density of soil, and $c = 0.2 \text{ cal g}^{-1} \text{ K}^{-1}$ is the specific thermal capacitance of soil.

The surface air temperature has been recorded as a function of time:

t [hours]	u °C
0	6
6	16
12	28
18	21
24	18
36	34
48	18
60	25
66	15
72	4

[H6.1b] Heat Diffusion in the Soil III

We want to assume that the surface soil temperature is identical with the surface air temperature at all times. We want to furthermore assume that the initial soil temperature is equal to the initial surface temperature everywhere.

[H6.1b] Heat Diffusion in the Soil III

We want to assume that the surface soil temperature is identical with the surface air temperature at all times. We want to furthermore assume that the initial soil temperature is equal to the initial surface temperature everywhere.

Specify the model using hours as units of time, and centimeters as units of space. Discretize the problem using third-order accurate finite differences everywhere.

[H6.1b] Heat Diffusion in the Soil III

We want to assume that the surface soil temperature is identical with the surface air temperature at all times. We want to furthermore assume that the initial soil temperature is equal to the initial surface temperature everywhere.

Specify the model using hours as units of time, and centimeters as units of space. Discretize the problem using third-order accurate finite differences everywhere.

Simulate the resulting linear ODE system using MATLAB's built-in stiff system solver (ode15s).

[H6.1b] Heat Diffusion in the Soil III

We want to assume that the surface soil temperature is identical with the surface air temperature at all times. We want to furthermore assume that the initial soil temperature is equal to the initial surface temperature everywhere.

Specify the model using hours as units of time, and centimeters as units of space. Discretize the problem using third-order accurate finite differences everywhere.

Simulate the resulting linear ODE system using MATLAB's built-in stiff system solver (ode15s).

Plot on one graph the soil temperature at the surface and at the insulator as functions of time.

[H6.1b] Heat Diffusion in the Soil III

We want to assume that the surface soil temperature is identical with the surface air temperature at all times. We want to furthermore assume that the initial soil temperature is equal to the initial surface temperature everywhere.

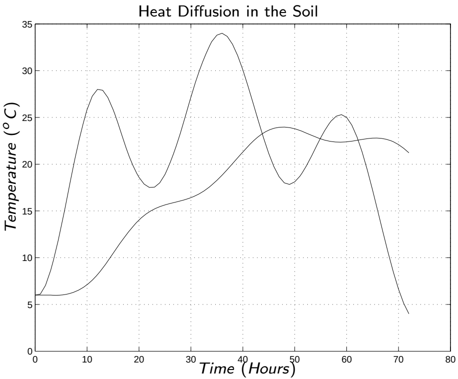
Specify the model using hours as units of time, and centimeters as units of space. Discretize the problem using third-order accurate finite differences everywhere.

Simulate the resulting linear ODE system using MATLAB's built-in stiff system solver (ode15s).

Plot on one graph the soil temperature at the surface and at the insulator as functions of time.

Generate also a three-dimensional plot showing the temperature distribution in the soil as a function of time and space.

[H6.1b] Heat Diffusion in the Soil IV



[H6.1b] Heat Diffusion in the Soil V