Hooke-Jeeves pattern search method

Define "norm" to measure length of vectors. Also set up routine to print given number of decimal places.

In[79]:=
```
Clear[norm, nn, dp, x];
norm[x_] = Sqrt[x.x];
dp = 6;
nn[x_] := NumberForm[N[x], {20, dp}];
```

Our standard function, gradient, etc. (actually don't need the Hessian for steepest descent). Added constant 2 to f so values will be positive; makes it easier to compare them.

In[83]:=
```
Clear[f, g, h, x, x1, x2, x3, xi];
f[{x1_, x2_, x3_}] = x1^2 + 2 x2^2 + 3 x3^2 - 1 / (x3 ^2 + 1) + Sin[x1 + 0.9 * x2 + 0.8 * x3] + 2
x = {x1, x2, x3};
g[{x1_, x2_, x3_}] = Map[Function[xi, D[f[x], xi]], x] ;
g[x] // MatrixForm
h[{x1_, x2_, x3_}] = Map[Function[xi, D[g[x], xi]], x] ;
h[x] // MatrixForm
```

Out[84]= $2 + x1^2 + 2\,x2^2 + 3\,x3^2 - \frac{1}{1 + x3^2} + \mathrm{Sin}[x1 + 0.9\,x2 + 0.8\,x3]$

Out[87]//MatrixForm=

$$\begin{pmatrix} 2\,x1 + \mathrm{Cos}[x1 + 0.9\,x2 + 0.8\,x3] \\ 4\,x2 + 0.9\,\mathrm{Cos}[x1 + 0.9\,x2 + 0.8\,x3] \\ 6\,x3 + \frac{2\,x3}{(1+x3^2)^2} + 0.8\,\mathrm{Cos}[x1 + 0.9\,x2 + 0.8\,x3] \end{pmatrix}$$

Out[89]//MatrixForm=

$$\begin{pmatrix} 2 - \mathrm{Sin}[x1 + 0.9\,x2 + 0.8\,x3] & -0.9\,\mathrm{Sin}[x1 + 0.9\,x2 + 0.8\,x3] & -0.8\,\mathrm{Sin}[x1 + 0.9\,x2 + 0.8\,x3] \\ -0.9\,\mathrm{Sin}[x1 + 0.9\,x2 + 0.8\,x3] & 4 - 0.81\,\mathrm{Sin}[x1 + 0.9\,x2 + 0.8\,x3] & -0.72\,\mathrm{Sin}[x1 + 0.9\,x2 + 0.8\,x3] \\ -0.8\,\mathrm{Sin}[x1 + 0.9\,x2 + 0.8\,x3] & -0.72\,\mathrm{Sin}[x1 + 0.9\,x2 + 0.8\,x3] & 6 - \frac{8\,x3^2}{(1+x3^2)^3} + \frac{2}{(1+x3^2)^2} - 0.64\,\mathrm{Sin}[x1 + 0.9 \end{pmatrix}$$

How to use the Hooke-Jeeves method stuff:

hjinit[initial x, initial h]

At each iteration

* begin with "pattern", then possibly "accept" if get better value
* if continuing, "base", then possibly "accept" if get better value
* if continuing, repeatedly "shrink" then possibly "accept" until "accept" or h is too small

On the first iteration, skip the "pattern" step. Once you have "accept"ed a value, start the next iteration.

In[90]:=
```
Clear[hjinit, x0, h0, explore, xx, printxtry, printxbase, pattern, base, shrink, accept];

hjinit[x0_, h0_] := Module[{},
      Clear[x, h, k, n, v, xtry];
      x[0] = N[x0]; h = N[h0]; n = Length[x0]; k = 0;
      v = IdentityMatrix[n];
      printxbase;
 ]

explore[xx_] := Module[{x},
      x = xx;
      For[i = 1, i ≤ n, i++,
           If[f[x + h * v[[i]] ] < f[x],
                x = x + h * v[[i]],
                If[f[x - h * v[[i]] ] < f[x],
                     x = x - h * v[[i]]
                 ]
           ]
       ];
      x
 ]

printxtry := Module[{},
      Print["  h = ", h // nn, "  trying x = ", xtry // nn, "  f = ", f[xtry] // nn];
```

```
    ]

printxbase := Module[{},
      Print[k, "  [", h // nn, "]   ", x[k] // nn, "  ", f[x[k]] // nn ];
    ]

pattern := Module[{},
      xtry = 2 * x[k] - x[k - 1];
      Print["  pattern jump to x = ", xtry // nn, "  f = ", f[xtry] // nn];
      xtry = explore[xtry];
      printxtry;
    ]

base := Module[{},
      xtry = x[k];
      xtry = explore[xtry];
      printxtry;
    ]

shrink := Module[{},
      h = h / 2;
      base;
    ]

accept := Module[{},
      x[k + 1] = xtry;
      k = k + 1;
      printxbase;
    ]
```

In[99]:= `hjinit[{0, 0, 0}, 1]`

```
0  [1.000000]   {0.000000, 0.000000, 0.000000}  1.000000
```

In[100]:= `base`

```
  h = 1.000000  trying x = {0.000000, 0.000000, 0.000000}  f = 1.000000
```

In[101]:= `shrink`

```
  h = 0.500000  trying x = {-0.500000, 0.000000, 0.000000}  f = 0.770574
```

In[102]:= `accept`

```
1  [0.500000]   {-0.500000, 0.000000, 0.000000}  0.770574
```

In[103]:= `pattern`

```
  pattern jump to x = {-1.000000, 0.000000, 0.000000}  f = 1.158529

  h = 0.500000  trying x = {-0.500000, 0.000000, 0.000000}  f = 0.770574
```

In[104]:= `base`

```
  h = 0.500000  trying x = {-0.500000, 0.000000, 0.000000}  f = 0.770574
```

In[105]:= **shrink**

```
h = 0.250000  trying x = {-0.500000, -0.250000, 0.000000}  f = 0.711865
```

In[106]:= **accept**

```
2  [0.250000]   {-0.500000, -0.250000, 0.000000}  0.711865
```

In[107]:= **pattern**

```
pattern jump to x = {-0.500000, -0.500000, 0.000000}  f = 0.936584
h = 0.250000  trying x = {-0.250000, -0.250000, 0.000000}  f = 0.730162
```

In[108]:= **base**

```
h = 0.250000  trying x = {-0.500000, -0.250000, 0.000000}  f = 0.711865
```

In[109]:= **shrink**

```
h = 0.125000  trying x = {-0.375000, -0.250000, -0.125000}  f = 0.683667
```

In[110]:= **accept**

```
3  [0.125000]   {-0.375000, -0.250000, -0.125000}  0.683667
```

In[111]:= **pattern**

```
pattern jump to x = {-0.250000, -0.250000, -0.250000}  f = 0.808926
h = 0.125000  trying x = {-0.375000, -0.125000, -0.125000}  f = 0.679853
```

In[112]:= **accept**

```
4  [0.125000]   {-0.375000, -0.125000, -0.125000}  0.679853
```

In[113]:= **pattern**

```
pattern jump to x = {-0.375000, 0.000000, -0.125000}  f = 0.745546
h = 0.125000  trying x = {-0.375000, -0.125000, -0.125000}  f = 0.679853
```

In[114]:= **base**

```
h = 0.125000  trying x = {-0.375000, -0.125000, -0.125000}  f = 0.679853
```

In[115]:= **shrink**

```
h = 0.062500  trying x = {-0.437500, -0.187500, -0.062500}  f = 0.667178
```

In[116]:= **accept**

```
5  [0.062500]   {-0.437500, -0.187500, -0.062500}  0.667178
```

In[117]:= **pattern**

```
pattern jump to x = {-0.500000, -0.250000, 0.000000}  f = 0.711865
h = 0.062500  trying x = {-0.437500, -0.187500, -0.062500}  f = 0.667178
```

In[118]:= **base**

```
h = 0.062500  trying x = {-0.375000, -0.187500, -0.062500}  f = 0.667074
```

```
In[119]:= dp = 9; accept

6  [0.062500000]   {-0.375000000, -0.187500000, -0.062500000}  0.667074169

In[120]:= pattern

  pattern jump to x = {-0.312500000, -0.187500000, -0.062500000}  f = 0.676967096

  h = 0.062500000  trying x = {-0.375000000, -0.187500000, -0.062500000}  f = 0.667074169

In[121]:= base

  h = 0.062500000  trying x = {-0.375000000, -0.187500000, -0.062500000}  f = 0.667074169

In[122]:= shrink

  h = 0.031250000  trying x = {-0.406250000, -0.187500000, -0.093750000}  f = 0.665244832

In[123]:= accept

7  [0.031250000]   {-0.406250000, -0.187500000, -0.093750000}  0.665244832

In[124]:= pattern

  pattern jump to x = {-0.437500000, -0.187500000, -0.125000000}  f = 0.674993028

  h = 0.031250000  trying x = {-0.406250000, -0.187500000, -0.093750000}  f = 0.665244832

In[125]:= base

  h = 0.031250000  trying x = {-0.406250000, -0.187500000, -0.093750000}  f = 0.665244832

In[126]:= shrink

  h = 0.015625000  trying x = {-0.406250000, -0.187500000, -0.078125000}  f = 0.664540260

In[127]:= accept

8  [0.015625000]   {-0.406250000, -0.187500000, -0.078125000}  0.664540260

In[128]:= pattern

  pattern jump to x = {-0.406250000, -0.187500000, -0.062500000}  f = 0.665864090

  h = 0.015625000  trying x = {-0.406250000, -0.187500000, -0.078125000}  f = 0.664540260

In[129]:= base

  h = 0.015625000  trying x = {-0.406250000, -0.187500000, -0.078125000}  f = 0.664540260

In[130]:= shrink

  h = 0.007812500  trying x = {-0.406250000, -0.179687500, -0.078125000}  f = 0.664467837

In[131]:= accept

9  [0.007812500]   {-0.406250000, -0.179687500, -0.078125000}  0.664467837

In[132]:= pattern

  pattern jump to x = {-0.406250000, -0.171875000, -0.078125000}  f = 0.664668700

  h = 0.007812500  trying x = {-0.406250000, -0.179687500, -0.078125000}  f = 0.664467837
```

```
In[133]:= base
  h = 0.007812500   trying x = {-0.406250000, -0.179687500, -0.078125000}   f = 0.664467837
In[134]:= shrink
  h = 0.003906250   trying x = {-0.406250000, -0.179687500, -0.082031250}   f = 0.664440987
In[135]:= accept
10  [0.003906250]   {-0.406250000, -0.179687500, -0.082031250}  0.664440987
```

More or less correct to 2 decimal places at this point - compare to Newton's method result of {-0.403510957,-0.181579931,-0.080964981}.

▪ Example from Walsh, p. 77

```
In[136]:= f[{x1_, x2_}] = 3 x1^2 - 2 x1 x2 + x2^2 + 4 x1 + 3 x2
```

Out[136]= $4\,x1 + 3\,x1^2 + 3\,x2 - 2\,x1\,x2 + x2^2$

```
In[137]:= hjinit[{0, 0}, 1]
0  [1.000000000]   {0.000000000, 0.000000000}  0.000000000
In[138]:= base
  h = 1.000000000   trying x = {-1.000000000, -1.000000000}   f = -5.000000000
In[139]:= accept
1  [1.000000000]   {-1.000000000, -1.000000000}  -5.000000000
In[140]:= pattern
  pattern jump to x = {-2.000000000, -2.000000000}   f = -6.000000000
  h = 1.000000000   trying x = {-1.000000000, -2.000000000}   f = -7.000000000
In[141]:= accept
2  [1.000000000]   {-1.000000000, -2.000000000}  -7.000000000
In[142]:= pattern
  pattern jump to x = {-1.000000000, -3.000000000}   f = -7.000000000
  h = 1.000000000   trying x = {-2.000000000, -3.000000000}   f = -8.000000000
In[143]:= accept
3  [1.000000000]   {-2.000000000, -3.000000000}  -8.000000000
In[144]:= pattern
  pattern jump to x = {-3.000000000, -4.000000000}   f = -5.000000000
  h = 1.000000000   trying x = {-2.000000000, -4.000000000}   f = -8.000000000
In[145]:= base
  h = 1.000000000   trying x = {-2.000000000, -3.000000000}   f = -8.000000000
```

In[146]:= **shrink**

```
  h = 0.500000000  trying x = {-1.500000000, -3.000000000}  f = -8.250000000
```

In[147]:= **accept**

```
4  [0.500000000]   {-1.500000000, -3.000000000}  -8.250000000
```

In[148]:= **pattern**

```
  pattern jump to x = {-1.000000000, -3.000000000}  f = -7.000000000
  h = 0.500000000  trying x = {-1.500000000, -3.000000000}  f = -8.250000000
```

In[149]:= **base**

```
  h = 0.500000000  trying x = {-1.500000000, -3.000000000}  f = -8.250000000
```

In[150]:= **shrink**

```
  h = 0.250000000  trying x = {-1.750000000, -3.250000000}  f = -8.375000000
```

In[151]:= **accept**

```
5  [0.250000000]   {-1.750000000, -3.250000000}  -8.375000000
```

In[152]:= **pattern**

```
  pattern jump to x = {-2.000000000, -3.500000000}  f = -8.250000000
  h = 0.250000000  trying x = {-1.750000000, -3.250000000}  f = -8.375000000
```

In[153]:= **base**

```
  h = 0.250000000  trying x = {-1.750000000, -3.250000000}  f = -8.375000000
```

In[154]:= **shrink**

```
  h = 0.125000000  trying x = {-1.750000000, -3.250000000}  f = -8.375000000
```

Stop since h = 1/8. Agrees with calculations in Walsh.