```
>
>
Digits := 20 :

# Your fixed-point function g
g := x → (x^3 − 0.5*x − 1.7) / sqrt(x^2 + 24) :

# Steffensen update S for g
S := x → x − ( (g(x) − x)^2) / ( g(g(x)) − 2*g(x) + x ) :

# Initial value and how many g-iterations per row
x0 := −1.3 :
m  := 3 :              # number of g-iterations to show per seed
Smax := 2 :              # we want rows for 0, 1, and 2 Steffensen updates

# Compute the row seeds:
# Base[0] = x0
# Base[1] = S(x0)
# Base[2] = S(S(x0))
Base := Array(0 ..Smax) :
Base[0] := evalf(x0) :
for s from 1 to Smax do
   Base[s] := evalf( S(Base[s−1]) ) :
od:

# Fill the results table X[#Steffensen_updates, #g-iterations]
# Column 0 is the seed; columns 1..m are successive g-iterates
X := Array(0 ..Smax, 0 ..m) :
for s from 0 to Smax do
   X[s, 0] := Base[s] :
   x := Base[s] :
   for k from 1 to m do
      x := evalf( g(x) ) :
      X[s, k] := x :
   od:
od:

# Pretty print
printf("X[#Steffensen_updates, #g-iterations]\n") :
for s from 0 to Smax do
   printf("Row s=%d:\n", s) :
   for k from 0 to m do
      printf("  X[%d,%d] = %.12f\n", s, k, X[s, k]) :
        od:
    od;
X[#Steffensen_updates, #g-iterations]
Row s=0:
  X[0,0] = -1.300000000000
  X[0,1] = -0.640619621473
  X[0,2] = -0.332463103524
  X[0,3] = -0.319844595634
Row s=1:
  X[1,0] = -0.062092977194
  X[1,1] = -0.340695211837
  X[1,2] = -0.319539422045
  X[1,3] = -0.320377309057
Row s=2:
  X[2,0] = -0.321032517826
  X[2,1] = -0.320312484540
  X[2,2] = -0.320343640053
  X[2,3] = -0.320342287893

>
```