1D Model Problem

Construct the typical tridiagonal Matrix for the 1D Poisson equation with zero right hand side vector b:

\[ n := 7; \]
\[
A := \text{Matrix}( \begin{array}{cccccccc}
2 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1 & 2 & -1 & 0 & 0 & 0 & 0 & 0 \\
0 & -1 & 2 & -1 & 0 & 0 & 0 & 0 \\
\end{array}, \text{shape}=\text{band}[1,1], \text{scan}=\text{band}[1,2]); \\
\text{evalf(Eigenvalues(A))};
\]

Conjugate Gradients
Standard CG-algorithm without preconditioning:

```maple
> CG := proc(A::Matrix, b::Vector, xstart::Vector, it::posint)
    #option trace;
    local i, alpha, beta, x, r, resnorm, d:
    x[0] := xstart;
    r := b - A.x[0];
    d := r;
    for i from 1 to it do
        resnorm := DotProduct(r, r);
        alpha := resnorm / BilinearForm(d, d, A);
        x[i] := x[i-1] + alpha * d;
        r := r - alpha * A.d;
        beta := DotProduct(r, r) / resnorm;
        d := r + beta * d;
    end do;
    return x;
end proc:

> xstart := evalf(RandomVector(n));
    # try an eigenvector instead:
    # k := 5;
    # xstart := < seq( evalf(sin(k*Pi*j/(n+1))), j=1..n) >;

xstart :=

| 27.  |
|  8.  |
| 69.  |
| 99.  |
| 29.  |
| 44.  |
| 92.  |

(2.1)

> iter := 20;
> xs := CG(A, b, xstart, iter);

iter := 20
xs := x

(2.2)

> listplot([ 0, seq(xs[6][i], i=1..n), 0]);
```
\[ \text{solplot := \{ seq( listplot([ 0, seq(xs[j][i], i=1..n), 0]), j=0..\text{iter} ) \};} \]
\[ \text{display(solplot, insequence=true);} \]
plot the maximum errors of each iterative solution:

> listplot( [ seq( Norm(xs[j]), j=0..iter ) ]);
Preconditioned Conjugate Gradient - Hierarchical Basis Preconditioning

Hierarchical Basis Transformation

hierarch corresponds to the inverse of the matrix $L^T$ of the lecture slides
(transforms coefficients for nodal basis into coefficients for hierarchical basis)

```maple
> hierarch := proc( x::Vector, n::posint )
  # option trace;
  local l,i, left, right, xx;
  xx := evalf(x);
  l := 1;
  while l < n/2 do
    for i from l to n by 2*l do
      if i-l>0 then left := xx[i-l] else left:=0 end if;
      if i+l<=n then right := xx[i+l] else right:=0 end if;
      xx[i] := xx[i]-(left+right)/2;
    end do;
```
dehierarch \text{corresponds to matrix } L^T \text{ on the lecture slides}
\text{(transforms coefficients for hierarchical basis into coefficients for nodal basis)}

> \text{dehierarch := proc} (x::Vector, n::posint)
  \#option trace;
  local l,i,left,right,xx;
  xx := evalf(x);
  l := (n+1)/4;
  while l >= 1 do
    for i from l to n by 2*l do
      if i-l>0 then left := xx[i-l] else left:=0 end if;
      if i+l<=n then right := xx[i+l] else right:=0 end if;
      xx[i] := xx[i] + (left+right)/2;
    end do;
    l := l/2;
  end do;
  return xx;
end proc:

hierarchBas \text{corresponds to the matrix } L
\text{(applied to a vector of nodal basis functions, it will transform this into a vector of hierarchical basis functions)}

> \text{hierarchBas := proc} (x::Vector, n::posint)
  \#option trace;
  local l,i,left,right,xx;
  xx := evalf(x);
  l := 1;
  while l <= (n+1)/4 do
    for i from 2*l to n by 2*l do
      xx[i] := xx[i] + (xx[i-l] + xx[i+l])/2;
    end do;
    l := l*2;
  end do;
  return xx;
end proc:

\text{Example for hierarchization - use a sin-mode as example:}

> k := 1;
  xstart := \text{seq} (evalf(sin(k*Pi*j/(n+1))), j=1..n);
  \text{k} := 1
CG with Hierarchical-Basis Preconditioning

```maple
PCG := proc(n::posint, A::Matrix, b::Vector, xstart::Vector, it::posint)
    option trace;
    local i, alpha, beta, x, r, resnorm, d, Ad:
    x[0] := xstart;
    r := hierarchBas(b - A.x[0], n);
    d := dehierarch(r, n);
    for i from 1 to it do
        resnorm := DotProduct(r, r);
        Ad := A.d;
        alpha := resnorm/DotProduct(d, Ad);
        x[i] := x[i-1] + alpha*d;
        r := r - alpha*hierarchBas(Ad, n);
        beta := DotProduct(r, r)/resnorm;
        d := dehierarch(r, n) + beta*d;
    end do;
    x
end proc:
```

**Example**

```maple
> xstart := Matrix([[0.3826834325], [0.7071067810], [0.9238795325], [1.0], [0.9238795325], [0.7071067810], [0.3826834325]]);

> hierarch(xstart, n);

> dehierarch(%);`
end do;
return x;
end proc:

> n := 7;
A := Matrix( [ [ seq(-1, i=1..n-1) ],
   [ seq(2, i=1..n) ],
   [ seq(-1, i=1..n-1) ] ], shape=band[1,1],
scan=band[1,2]);
b := < seq(0, i=1..n) >;
xstart := evalf( RandomVector(n) );
# k := 3;
# xstart := < seq( evalf(sin(k*Pi*j/(n+1))), j=1..n) >;

iter := 10;
xs := PCG(n,A,b,xstart,iter);
solplot := [ seq( listplot([ 0, seq(xs[j][i], i=1..n), 0 ]), j=0..iter) ]:
display(solplot, insequence=true);

\[ (3.2.1) \]
Plot the solution after the first iteration -> basically =0 (hence, convergence in a single step).

> listplot([ 0, seq(xs[3][i], i=1..n), 0]);
**Explanation: Compute the Preconditioning Matrices**

The matrix $L$ (for hierarchization of the nodal basis) can be computed column-wise by applying the transform hierarchBas to all unit vectors. $\text{transpose}(L)$ is then the dehierarchization of the hierarchical coefficients. In $\text{Linv}$, we will compute the inverse of $L$ - $\text{transpose}(\text{Linv})$ is the inverse of $\text{transpose}(L)$, resp.

Note that in Maple notation $<a,b,c>$ denotes a matrix, if $a$, $b$, and $c$ are row vectors of the matrix.

```maple
n := 7;
A := Matrix( 
[ 
seq(-1, i=1..n-1),
seq( 2, i=1..n),
seq(-1, i=1..n-1) 
], shape=band[1,1],
scan=band[1,2]);
```

\[
A := \begin{bmatrix}
2 & -1 & 0 & 0 & 0 & 0 & 0 \\
-1 & 2 & -1 & 0 & 0 & 0 & 0 \\
0 & -1 & 2 & -1 & 0 & 0 & 0 \\
0 & 0 & -1 & 2 & -1 & 0 & 0 \\
0 & 0 & 0 & -1 & 2 & -1 \\
0 & 0 & 0 & 0 & -1 & 2 \\
0 & 0 & 0 & 0 & 0 & -1 & 2
\end{bmatrix}
\]

(3.3.1)
\[ \text{L} := \text{seq} (\text{Transpose( dehierarch( \text{seq}(\text{`if`(i=j,1,0),j=1..n} >,n) ), i=1..n) >}) ; \]
\[
L := \begin{bmatrix}
1., 0., 0., 0., 0., 0., 0. \\
0.5000000000, 1., 0.5000000000, 0., 0., 0., 0. \\
0., 0., 1., 0., 0., 0., 0. \\
0.2500000000, 0.5000000000, 0.7500000000, 1., 0.7500000000, 0.5000000000, 0.2500000000, 0., 0., 0., 0., 1., 0. \\
0., 0., 0., 0., 0.5000000000, 0.5000000000, 1., 0.5000000000, 0. \\
0., 0., 0., 0., 0., 0., 0. \\
\end{bmatrix}
\]

\[ \text{Linv} := \text{Transpose( \text{seq}(\text{Transpose( hierarch( \text{seq}(\text{`if`(i=j,1,0),j=1..n} >,n) ), i=1..n) >}) ;} \]
\[
\text{Linv} := \begin{bmatrix}
1. 0.5000000000 0. 0. 0. 0. 0. 0. 0. \\
0. 1. 0. 0. 0. 0. 0. 0. 0. \\
0. 0. 1. 0. 0. 0. 0. 0. 0. \\
0. 0. 0. 0. 0. 0. 0. 0. 0. \\
0. 0. 0. 0. 0. 0. 0. 0. 0. \\
0. 0. 0. 0. 0. 0. 0. 0. 0. \\
\end{bmatrix}
\]

\[ \text{LT := Transpose( \text{seq}(\text{Transpose( hierarchBas( \text{seq}(\text{`if`(i=j,1,0),j=1..n} >,n) ), i=1..n) >}) ;} \]
\[
\text{LT :=} \begin{bmatrix}
1. 0.5000000000 0. 0. 0. 0. 0. 0. 0. \\
0. 1. 0. 0. 0. 0. 0. 0. 0. \\
0. 0. 1. 0. 0. 0. 0. 0. 0. \\
0. 0. 0. 0. 0. 0. 0. 0. 0. \\
0. 0. 0. 0. 0. 0. 0. 0. 0. \\
0. 0. 0. 0. 0. 0. 0. 0. 0. \\
\end{bmatrix}
\]

Note that \( R \) and \( L \) are inverse to each other (as hierarchical "forward" and "backward" transformations)

\[ \text{L.Linv;} \]
\[
\begin{bmatrix}
1., 0.5000000000, 0., 0., 0., 0., 0., 0., 0. \\
0., 1., 0., 0., 0., 0., 0., 0., 0. \\
0., 0., 1., 0., 0., 0., 0., 0., 0. \\
0., 0., 0., 0., 0., 0., 0., 0., 0. \\
0., 0., 0., 0., 0., 0., 0., 0., 0. \\
0., 0., 0., 0., 0., 0., 0., 0., 0. \\
\end{bmatrix}
\]
Most important: $L^T A L$ is a diagonal matrix (which explains the rapid CG convergence); note that the size of the diagonal levels depends on the hierarchical level of the corresponding basis functions!

\[
LAL := \text{Transpose}(L) \cdot \text{Matrix}(A) \cdot L;
\]

\[
LAL := \left[
\begin{array}{cccc}
1.625000000000000 & 0.250000000000000 & 0.375000000000000 & 0.125000000000000 \\
0.250000000000000 & 2.500000000000000 & 0.250000000000000 & 0.500000000000000 \\
0.250000000000000 & 0.500000000000000 & 0.250000000000000 & 1.125000000000000 \\
0.500000000000000 & 0.500000000000000 & 0.125000000000000 & 0.500000000000000 \\
0.500000000000000 & 0.500000000000000 & 1.000000000000000 & 0.500000000000000
\end{array}
\right]
\]

\[
b := \langle \text{seq}(0, \ i=1..n) \rangle;
\]

\[
xstart := \text{evalf}(\text{RandomVector}(n));
\]

\[
b := \left[
\begin{array}{c}
0 \\
0 \\
-18. \\
87. \\
33. \\
0 \\
0 \\
0 \\
-98. \\
-77. \\
57. \\
27.
\end{array}
\right]
\]
Apply regular CG algorithm to the transformed system of equations; note that we transform the starting solution $x_{\text{start}}$ into hierarchical-basis form:

```maple
t := 3;
x := CG(LAL, hierarchBas(b, n), hierarch(xstart, n), iter);
solplot := [ seq( listplot([ 0, seq(xs[j][i], i=1..n), 0 ]), j=0..iter ) ];
display(solplot, insequence=true);
```

```
iter := 3
xs := x
```