```
> restart;
> with(LinearAlgebra);
```

[*&x*, *Add*, *Adjoint*, *BackwardSubstitute*, *BandMatrix*, *Basis*, *BezoutMatrix*, *BidiagonalForm*, *BilinearForm*, *CharacteristicMatrix*, *CharacteristicPolynomial*, *Column*, *ColumnDimension*, *ColumnOperation*, *ColumnSpace*, *CompanionMatrix*, *ConditionNumber*, *ConstantMatrix*, *ConstantVector*, *Copy*, *CreatePermutation*, *CrossProduct*, *DeleteColumn*, *DeleteRow*, *Determinant*, *Diagonal*, *DiagonalMatrix*, *Dimension*, *Dimensions*, *DotProduct*, *EigenConditionNumbers*, *Eigenvalues*, *Eigenvectors*, *Equal*, *ForwardSubstitute*, *FrobeniusForm*, *GaussianElimination*, *GenerateEquations*, *GenerateMatrix*, *Generic*, *GetResultDataType*, *GetResultShape*, *GivensRotationMatrix*, *GramSchmidt*, *HankelMatrix*, *HermiteForm*, *HermitianTranspose*, *HessenbergForm*, *HilbertMatrix*, *HouseholderMatrix*, *IdentityMatrix*, *IntersectionBasis*, *IsDefinite*, *IsOrthogonal*, *IsSimilar*, *IsUnitary*, *JordanBlockMatrix*, *JordanForm*, *LA_Main*, *LUDecomposition*, *LeastSquares*, *LinearSolve*, *Map*, *Map2*, *MatrixAdd*, *MatrixExponential*, *MatrixFunction*, *MatrixInverse*, *MatrixMatrixMultiply*, *MatrixNorm*, *MatrixPower*, *MatrixScalarMultiply*, *MatrixVectorMultiply*, *MinimalPolynomial*, *Minor*, *Modular*, *Multiply*, *NoUserValue*, *Norm*, *Normalize*, *NullSpace*, *OuterProductMatrix*, *Permanent*, *Pivot*, *PopovForm*, *QRDecomposition*, *RandomMatrix*, *RandomVector*, *Rank*, *RationalCanonicalForm*, *ReducedRowEchelonForm*, *Row*, *RowDimension*, *RowOperation*, *RowSpace*, *ScalarMatrix*, *ScalarMultiply*, *ScalarVector*, *SchurForm*, *SingularValues*, *SmithForm*, *StronglyConnectedBlocks*, *SubMatrix*, *SubVector*, *SumBasis*, *SylvesterMatrix*, *ToeplitzMatrix*, *Trace*, *Transpose*, *TridiagonalForm*, *UnitVector*, *VandermondeMatrix*, *VectorAdd*, *VectorAngle*, *VectorMatrixMultiply*, *VectorNorm*, *VectorScalarMultiply*, *ZeroMatrix*, *ZeroVector*, *Zip*] **(1)**

Décomposition QR d'une matrice réelle

Gram-Schmidt

```
> gs:=proc(a)
  local r,b,i,j,k,l,n,c,s,m;
  n:=ColumnDimension(a);
  m:=Vector(n,j->a[1..n,j]);
  r:=Matrix(n,n,0);
  for i from 1 to n do
      for j from 1 to i-1 do
          r[j,i]:=evalf(m[i].b[j]);
      od;
      c[i]:=m[i];
      for j from 1 to i-1 do
           c[i]:=evalf(c[i]-r[j,i]*b[j]);
      od;
      r[i,i]:=evalf(VectorNorm(c[i],2));
      b[i]:=c[i]/r[i,i];
  od;
  [r,Matrix(n,n,(k,l)->b[l][k])];
  end;
```

```
gs := proc(a)
    local r, b, i, j, k, l, n, c, s, m,
    n := LinearAlgebra:-ColumnDimension(a);
    m := Vector(n, j -> a[1..n, j]);
    r := Matrix(n, n, 0);
    for i to n do
        for j to i - 1 do r[j, i] := evalf((`.`)(m[i], b[j])) end do;
        c[i] := m[i];
        for j to i - 1 do c[i] := evalf(c[i] - r[j, i] * b[j]) end do;
        r[i, i] := evalf(LinearAlgebra:-VectorNorm(c[i], 2));
        b[i] := c[i] / r[i, i]
    end do;
    [r, Matrix(n, n, (k, l) -> b[l][k])]
end proc
```

(1.1)

```
> m:=<<1,2,3>|<1,0,1>|<0,2,1>>;
```

$$m := \begin{bmatrix} 1 & 1 & 0 \\ 2 & 0 & 2 \\ 3 & 1 & 1 \end{bmatrix} \tag{1.2}$$

```
> dec:=gs(m);
```

$$dec := \left[\begin{bmatrix} 3.741657387 & 1.069044968 & 1.870828693 \\ 0 & 0.9258200998 & -1.080123451 \\ 0 & 0 & 0.5773502690 \end{bmatrix}, \right. \tag{1.3}$$

$$\left. \begin{bmatrix} 0.267261241900000001 & 0.771516749907418142 & 0.577350271354059252 \\ 0.534522483800000003 & -.617213400185164129 & 0.577350267370342340 \\ 0.801783725700000005 & 0.154303349722253958 & -.577350268755983053 \end{bmatrix} \right]$$

```
> %[2].%[1];
```

$$\begin{bmatrix} 1.00000000001392886 & 1.00000000019128344 & -6.74814648604638024\ 10^{-11} \\ 2.00000000002785772 & -1.68074443251953198\ 10^{-10} & 1.99999999973160026 \\ 3.00000000004178702 & 1.00000000002320876 & 1.00000000004365152 \end{bmatrix} \tag{1.4}$$

```
> P:=Transpose(dec[2]).dec[2];
```

$$P := [[0.99999999990702438, -3.46410206075731253\ 10^{-10}, -4.62909710563508270\ 10^{-11}], [-3.46410206075731253\ 10^{-10}, 1.00000000049134540, 2.85969534163932337\ 10^{-9}], [-4.62909710563508270\ 10^{-11}, 2.85969534163932337\ 10^{-9}, 0.99999999989781752]] \tag{1.5}$$

```
> Matrix(3,3,(i,j)->round(P[i,j]));
```

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (1.6)$$

>

Householder

```
> E1:=proc(n)
  Vector(n,i->if i=1 then 1 else 0 fi)
  end;
```

$$E1 := \mathbf{proc}(n)\ Vector(n, i \rightarrow \mathbf{if}\ i = 1\ \mathbf{then}\ 1\ \mathbf{else}\ 0\ \mathbf{end\ if})\ \mathbf{end\ proc} \quad (2.1)$$

```
> E1(4);
```

$$\begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad (2.2)$$

>

La fonction suivante calcule la matrice de Householder d'un vecteur X donné.

```
> HH:=proc(X)
  local W,n;
  n:=Dimension(X);
  W:=evalf(X-Norm(X,2)*E1(n));
  IdentityMatrix(n)-2/(Transpose(W).W)*W.Transpose(W)
  end;
```

$$HH := \mathbf{proc}(X) \quad (2.3)$$

$$\mathbf{local}\ W, n;$$

$$n := LinearAlgebra\text{-}Dimension(X);$$

$$W := evalf(X - LinearAlgebra\text{-}Norm(X, 2) * E1(n));$$

$$LinearAlgebra\text{-}IdentityMatrix(n) - `.`(2 * W / `.`(LinearAlgebra\text{-}Transpose(W), W), LinearAlgebra\text{-}Transpose(W))$$

$$\mathbf{end\ proc}$$

```
> HH(<1,2,3>);
```

$$\begin{bmatrix} 0.267261241891287970 & 0.534522483796194381 & 0.801783725694291571 \\ 0.534522483796194381 & 0.610073464080000072 & -.584889803880000003 \\ 0.801783725694291682 & -.584889803880000003 & 0.122665294179999940 \end{bmatrix} \quad (2.4)$$

```
> %.<1,2,3>;
```

$$\begin{bmatrix} 3.74165738656655122 \\ 3.16194403993108608\ 10^{-10} \\ 4.74291494967360450\ 10^{-10} \end{bmatrix} \quad (2.5)$$

>

La fonction suivante donne la décomposition QR, par la méthode de Householder.

Cet algorithme gagnerait à être optimisé ...

```
> QRHouseholder:=proc(m)
  local q,r,i,n,s;
  n:=ColumnDimension(m);
  r:=m;
  q:=IdentityMatrix(n);
  for i from 1 to n-1 do
      s:=HH(r[i..n,i]);
      s:=DiagonalMatrix([IdentityMatrix(i-1),s]);
      q:=s.q;
      r:=s.r;
  od;
  [r,Transpose(q)];
  end;
```

$$QRHouseholder := \mathbf{proc}(m) \tag{2.6}$$

```
    local q, r, i, n, s,
    n := LinearAlgebra:-ColumnDimension(m);
    r := m,
    q := LinearAlgebra:-IdentityMatrix(n);
    for i to n - 1 do
        s := HH(r[i..n, i]);
        s := LinearAlgebra:-DiagonalMatrix([LinearAlgebra:-IdentityMatrix(i - 1), s]);
        q := `.`(s, q);
        r := `.`(s, r)
    end do;
    [r, LinearAlgebra:-Transpose(q)]
end proc
```

```
> QR:=QRHouseholder(m);
```

$$QR := \tag{2.7}$$

$$\left[\left[\left[3.74165738656655122,\ 1.06904496758557954,\ 1.87082869328668044\right],\right.\right.$$

$$\left[4.56387199838824787\ 10^{-10},\ 0.925820099862187939,\ -1.08012344942700177\right],$$

$$\left[3.41528984331964941\ 10^{-10},\ -6.92640458876958576\ 10^{-11},\ 0.577350269544195216\right.$$

$$\left.\left.\left.\right]\right], \begin{bmatrix} 0.267261241891287970 & 0.771516749840986837 & 0.577350269064137999 \\ 0.534522483796194381 & -.617213399724101497 & 0.577350269338798627 \\ 0.801783725694291571 & 0.154303350021201046 & -.577350269133402038 \end{bmatrix}\right]$$

```
> %[2].%[1];
```

$$\begin{bmatrix} 1.00000000041477954 & 1.00000000001769718 & 2.70291400372713042\ 10^{-10} \\ 1.99999999969742404 & -4.51716925384964901\ 10^{-11} & 1.99999999985951748 \\ 2.99999999954613550 & 1.00000000001118728 & 0.99999999965055286 \end{bmatrix} \quad (2.8)$$

```
> m;
```

$$\begin{bmatrix} 1 & 1 & 0 \\ 2 & 0 & 2 \\ 3 & 1 & 1 \end{bmatrix} \quad (2.9)$$

```
>
> Transpose(QR[2]).QR[2];
```

$$[[0.99999999988914546,\ 1.16721160514643429\ 10^{-10},\ 8.73461858397206470\ 10^{-11}],\ [1.16721160514643429\ 10^{-10},\ 0.99999999991194854,\ -1.24968521642010444\ 10^{-10}],\ [8.73461858397206470\ 10^{-11},\ -1.24968521642010444\ 10^{-10},\ 0.99999999996242760]] \quad (2.10)$$

```
>
>
```

Fonction Maple

```
> QRDecomposition(m);
```

$$\begin{bmatrix} \frac{1}{14}\sqrt{14} & \frac{5}{42}\sqrt{42} & \frac{1}{3}\sqrt{3} \\ \frac{1}{7}\sqrt{14} & -\frac{2}{21}\sqrt{42} & \frac{1}{3}\sqrt{3} \\ \frac{3}{14}\sqrt{14} & \frac{1}{42}\sqrt{42} & -\frac{1}{3}\sqrt{3} \end{bmatrix}, \begin{bmatrix} \sqrt{14} & \frac{2}{7}\sqrt{14} & \frac{1}{2}\sqrt{14} \\ 0 & \frac{1}{7}\sqrt{42} & -\frac{1}{6}\sqrt{42} \\ 0 & 0 & \frac{1}{3}\sqrt{3} \end{bmatrix} \quad (3.1)$$

```
> %[1].%[2];
```

$$\begin{bmatrix} 1 & 1 & 0 \\ 2 & 0 & 2 \\ 3 & 1 & 1 \end{bmatrix} \quad (3.2)$$

```
>
```