La méthode de Daniel Ferand appliquée à un exemple de matrice 4x4

```
> restart;
with(linalg);
```

```
Warning, the protected names norm and trace have been redefined and unprotected
```

$[BlockDiagonal, GramSchmidt, JordanBlock, LUdecomp, QRdecomp, Wronskian, addcol, addrow, adj, adjoint, angle, augment, backsub, band, basis, bezout, blockmatrix, charmat, charpoly, cholesky, col, coldim, colspace, colspan, companion, concat, cond, copyinto, crossprod, curl, definite, delcols, delrows, det, diag, diverge, dotprod, eigenvals, eigenvalues, eigenvectors, eigenvects, entermatrix, equal, exponential, extend, ffgausselim, fibonacci, forwardsub, frobenius, gausselim, gaussjord, geneqns, genmatrix, grad, hadamard, hermite, hessian, hilbert, htranspose, ihermite, indexfunc, innerprod, intbasis, inverse, ismith, issimilar, iszero, jacobian, jordan, kernel, laplacian, leastsqrs, linsolve, matadd, matrix, minor, minpoly, mulcol, mulrow, multiply, norm, normalize, nullspace, orthog, permanent, pivot, potential, randmatrix, randvector, rank, ratform, row, rowdim, rowspace, rowspan, rref, scalarmul, singularvals, smith, stackmatrix, submatrix, subvector, sumbasis, swapcol, swaprow, sylvester, toeplitz, trace, transpose, vandermonde, vecpotent, vectdim, vector, wronskian]$

linalg est la bibliothèque de l'algèbre linéaire (nécessaire pour travailler sur les matrices)

1 - Entrée de la matrice A, à décomposer :

```
> A0 := matrix([
[1,1,3,-1],
[-1,1,2,0],
[0,0,1,2],
[0,0,0,1]
]);
A:=A0;
eigenvectors(A);
```

$$A0 := \begin{bmatrix} 1 & 1 & 3 & -1 \\ -1 & 1 & 2 & 0 \\ 0 & 0 & 1 & 2 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A := A0$$

$$[1+I, 1, \{[1, I, 0, 0]\}], [1-I, 1, \{[1, -I, 0, 0]\}], [1, 2, \{[2, -3, 1, 0]\}]$$

2 - Calcul du polynôme caractéristique ChiA de la matrice A et de sa dérivée :

```
> ChiA:=X->charpoly(A,X):
'ChiA(X)'= ChiA(X);
```

```
dChiA:=X->subs(D(X)=1,D(ChiA(X))):
'dChiA(X)'=dChiA(X);
```

$$\mathrm{ChiA}(X) = X^4 - 4\,X^3 + 7\,X^2 - 6\,X + 2$$

$$\mathrm{dChiA}(X) = 4\,X^3 - 12\,X^2 + 14\,X - 6$$

3 - Calcul du polynôme Q :

```
> divide (ChiA(X),gcd(ChiA(X),dChiA(X)),'Q'):
'Q'=Q;
dQ:=subs(D(X)=1,D(Q));
```

$$Q = X^3 - 3\,X^2 + 4\,X - 2$$

$$dQ := 3\,X^2 - 6\,X + 4$$

4 - Calcul du deuxième terme de la suite :

```
> A:=evalm(A-subs(X=A,Q) &* inverse(subs(X=A,subs(D(X)=1,dQ))));
```

$$A := \begin{bmatrix} 1 & 1 & 3 & -5 \\ -1 & 1 & 2 & 6 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

5 - Calcul des autres termes de la suite :

```
> p:=1:
B:=evalm(A0):
while rank(A-B) > 0 do
B:=evalm(A):
A:=evalm(A-subs(X=A,Q) &* inverse(subs(X=A,subs(D(X)=1,dQ))));
print(p,A=evalm(A));
print(`-----------------------------------------------------------------
-----------------`);
p:=p+1:
od:
N:=evalm(A0-A);
```

$$1,\, A = \begin{bmatrix} 1 & 1 & 3 & -5 \\ -1 & 1 & 2 & 6 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$N := \begin{bmatrix} 0 & 0 & 0 & 4 \\ 0 & 0 & 0 & -6 \\ 0 & 0 & 0 & 2 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Conclusion :
On constate que la suite se stabilise très vite (dès le rang 2) ce qui est conforme à la théorie qui permet de prévoir que la suite est stationnaire à partir du rang n si : $\dim(E) < 2^n$.
En fait, c'est même beaucoup mieux puisqu'il suffit de que 2^n dépasse r (où r est le plus grand ordre de multiplicité des valeurs propres). Ici r=3 donc la condition est : 2^n > 3, ce qui correspond à n >1.
On retrouve ainsi que la méthode de Newton à une rapidité de convergence "fabuleuse ou phénoménale" comme le dit Cédric Villani.