Quasi-Polynomial Mapping based Rootfinder

QPmR v.2
Created by Tomas Vyhlidal, CTU in Prague
http://www.cak.fs.cvut.cz/algorithms/qpmr

```
QPmR(Region,P,D)

   Finds all zeros of the quasi-polynomial

   QP(s)=(P(1,1)*s^n+...+P(1,n)*s+P(1,n+1))*exp(-D(1)*s)+
    +(P(2,1)*s^n+...+P(2,n)*s+P(2,n+1))*exp(-D(2)*s)+
                   ...
    +(P(N,1)*s^n+...+P(N,n)*s+P(N,n+1))*exp(-D(N))*s)

   located in the complex plane region defined by

   Region=[real_min real_max imag_min imag_max]

   The other two function inputs are:

   P  - N by n matrix of polynomial coefficients, n is the maximum power of s
        in the quasi-polynomial, N is the number of delays.
   D  - vector of different (positive) delay values of size N. One of the
        delays should be equal to 0.

   The quasi-polynomial can also be defined in the function handle form Fun.
   Then, the QPmR function syntax is

QPmR(Region,Fun)

   The QPmR function output is the vector of all quasi-polynomial zeros
located in
   the given region. The method is based on mapping the quasi-polynomial zero
   level curves over the complex plane region with the adaptation of the
mapping
   grid.

NaN as the function output indicates failure of the adaptation algorithm.
  In this case, the region Reg should be reduced. Alternatively, the grid can
  be assigned manually using the extended function modes

QPmR(Region,P,D,e,ds,gr) or
QPmR(Region,Fun,e,ds,gr)

  where the additional input parameters are

   e  - computation accuracy. If e=-1, then e=1e-6*ds (the same accuracy
        is considered if e parameter is not given, as it is above).
   ds - grid step for mapping the zero-level curves. If ds=-1, the grid
        step is adjusted automatically.
   gr - graphical representation of the results. If gr=1, results are
        visualized in plots (default gr=0). If the quasi-polynomial is
        given in P and D, the spectrum distribution diagram and the
        asymptotic exponentials of the spectra are visualized. If the
```

```
        quasi-polynomial is neutral, also the spectrum of the associated
        difference equation is computed and its safe upper bound is
        determined and visualized. If gr=1, the information on the grid
        adaptation is provided in the command window.

[R Y]=(Region,P,D,....) provides the following outputs

   R          - computed zeros of the quasi-polynomial (NaN indicates the
                algorithm failure)
   Y          - structure with summary of the QPmR results, particularly
   Y.zeros    - computed zeros of the quasi-polynomial (available also
                if R=NaN)
   Y.flag     - result correctness flag.
                Y.flag=1  - the positive result of cross-checking implies
                            that the zeros are computed correctly.
                Y.flag=0  - method failure: either the region is too
                            large or there are multiple or dense
                            roots. Next, the function can also be
                            ill-conditioned. Try to reduce the
                            region.
                Y.flag=-1 - method failure: too large grid, which causes
                            Newton's iterations failure. The grid size
                            ds should be reduced (manually, if needed).
   Y.accuracy - accuracy estimate of the computed zeros
   Y.asympt   - parameters [mi c] of the asymptotic exponentials of the
                root chains real=c(k)-mi(k)*log(imag);
   Y.function - quasi-polynomial in the function handle form
   Y.grid     - final grid size

   Additional outputs for neutral quasi-polynomials
   Y.DEzeros    - computed zeros of the associated difference equation
   Y.DEflag     - result correctness flag (the same as above)
   Y.DEaccuracy - accuracy estimate of the computed zeros
   Y.DEupbound  - safe upper bound on the spectrum of difference equation
   Y.DEfunction - difference equation in the function handle form
   Y.DEgrid     - final grid size


[R Y]=(Region,Fun,....) provides the following outputs

   R          - computed zeros of the quasi-polynomial (NaN indicates the
                algorithm failure)
   Y          - summary of the QPmR function results.
   Y.zeros    - computed zeros of the quasi-polynomial
   Y.flag     - result correctness flag - the same as above
   Y.accuracy - accuracy estimate of the computed zeros
   Y.grid     - final grid size

Remark 1: in the QPmR(Reg,Fun,...), the function can be used for computing
roots of general analytical functions, e.g. fractional polynomials or
quasi-polynomials.

Remark 2: in the automatic adjustment of the grid size (ds==-1), first,
the grid size is set to ds=(Reg(2)-Reg(1))*(Reg(4)-Reg(3))/Ns,
where Ns=1000 (or Ns=500 for more complex functions). If not sufficient,
the grid size is up to twice time reduced by the factor of four.
If not sufficient, the region is divided first to four and then up to 16
```

Sub-regions if needed, and the QPmR runs recursively in two recursion levels.

Example

```
   Find all the roots of the quasi-polynomial
   Q(s)=(1.5*s^3+0.2*s^2+20.1)+(s^3-2.1*s)*exp(-s*1.3)+
        +3.2*s*exp(-s*3.5)+1.4*exp(-s*4.3)
   located in the region Reg=[-10 5 0 300]
```

a) representation of the quasi-polynomial by the coefficient matrix and vectors of delays:

```
P=[1.5 0.2 0 20.1;1 0 -2.1 0;0 0 3.2 0;0 0 0 1.4]
D=[0;1.3;3.5;4.3]

R=QPmR([-10 5 0 300],P,D) - provides the computed zeros in the vector R.
   No graphical outputs, accuracy and grid size are adjusted automatically.

[R Y]=QPmR([-10 5 0 300],P,D,-1,-1,1) - provides the computed zeros in the
   vector R. Structure Y contains additional information on the spectrum
   and its computational aspects. With graphical outputs, accuracy
   and grid size are adjusted automatically.

[R Y]=QPmR([-10 5 0 300],P,D,1e-8) - with given accuracy 1e-8. Grid size
   is adjusted automatically, no graphical outputs.

[R Y]=QPmR([-10 5 0 300],P,D,1e-8,0.1) - with given accuracy 1e-8 and the
   fixed grid size 0.1. No graphical outputs. No grid adaptation.
```

b) representation of the quasi-polynomial by the function handle

```
   Fun=@(s)(1.5*s.^3+0.2*s.^2+20.1)+(s.^3-2.1.*s).*exp(-s*1.3)+
        3.2.*s.*exp(-s*3.5)+1.4.*exp(-s*4.3)

   R=QPmR([-10 5 0 300],Fun)

   [R Y]=QPmR([-10 5 0 300],P,D,-1,-1,1)
```