

Purpose Robust stability area for polynomials with parametric uncertainties

Syntax

```
S = sarea(q1,...,qm,ExpressionString,p0,p1,...,pn[,tol])

sareaplot(q1,S[,PlotType][,'new'])
sareaplot(q1,q2,S[,PlotType][,'new'])
sareaplot(q1,q2,q3,S[,PlotType][,'new'])
```

Description The command

```
S = sarea(q1,...,qm,ExpressionString,p0,p1,...,pn[,tol])
```

investigates robust stability for a family of polynomials depending on m uncertain parameters. By gridding of the parameter space, the family splits into a number of standard polynomials that are separately checked for stability. The m -dimensional grid is defined by the vectors of parameter values q_1, \dots, q_m and the results are stored into an m -dimensional array S structured accordingly. As expected, in S 1's stand for "stable" and 0's for "unstable". For details on a single polynomial stability checking, please read the description of macro `isstable`. The parameters p_0, \dots, p_n are given fixed polynomials serving to define the uncertainty structure. Beware that the input arguments representing both the parameters and the fixed polynomials must be written using their names (rather than values) in the function call. The uncertainty structure of the polynomial family is defined by the string variable `ExpressionString`. This string may contain any Matlab-like expression composed of the parameter names (acting here as scalars) and of the fixed polynomials names. The procedure is better explained in the examples below. For three or more uncertain parameters, dense gridding may result in slow performance. Typing `verbose yes` before the run can activate an on-line info on the macro performance.

Once the array S is ready, it can be plot by typing one of

```
sareaplot(q1,S[,PlotType][,'new'])
sareaplot(q1,q2,S[,PlotType][,'new'])
sareaplot(q1,q2,q3,S[,PlotType][,'new'])
```

for the case of one, two or three parameters, respectively. As above, the parameters q_1, q_2, q_3 must be typed by names and not by values. The optional argument `PlotType` specifies type of the plot. It can either be a surface plot (default or `PlotType='surf'`), point plot (`PlotType='points'`) or a combination of the two (`PlotType='both'`). The surface plot is typically nicer but can miss some details, while the point plot is always complete. With the input string argument `'new'`, the plot is displayed in a new window.

sarea, sareaplot

sarea, sareaplot-2

Examples

The following examples illustrate how the command should be used.

Example 1

Consider an uncertain polynomial

$$p(s, q_1, q_2) = p_0(s) + (q_1 + q_2)p_1(s) + \sqrt{|q_2|}p_2(s)$$

composed of three fixed polynomials

$$p_0 = 4 + 8s + 5s^2 + s^3$$

$$p_1 = 1 - s + s^2$$

$$p_2 = s + s^4$$

and two real parameters $q_1 \in [-6, 12]$ and $q_2 \in [-5, 15]$. Suppose you want to check which values of q_1 and q_2 give rise to a stable $p(s, q_1, q_2)$. As the parameters are two and the uncertainty structure is quite complicated, there is hardly any theoretical method known to help. Nevertheless, simple gridding can do the job in a reasonable time.

To start, insert the given data

```
p0=4+8*s+5*s^2+s^3; p1=1-s+s^2; p2=s+s^4;
```

and choose an appropriate gridding, such as

```
q1=-6:.1:12; q2=-5:.1:15;
```

Then construct the stability area array by typing

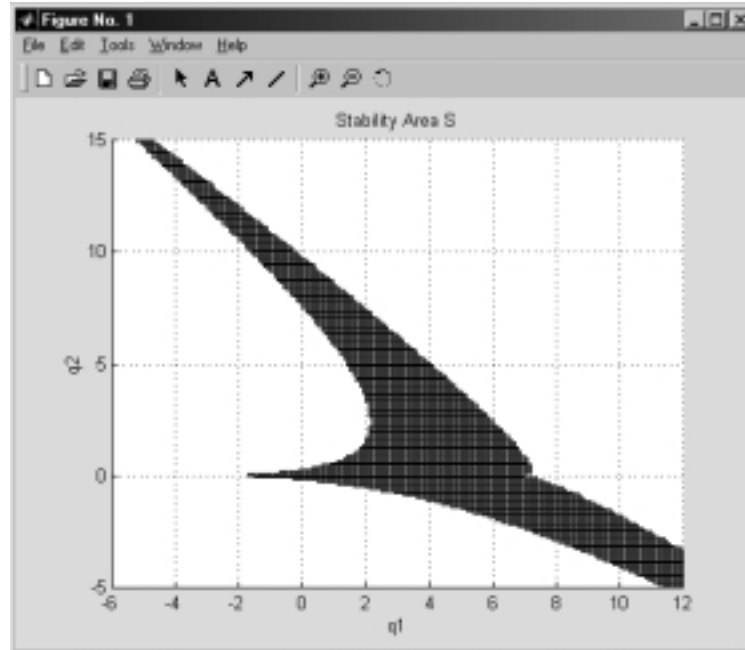
```
S=sarea(q1,q2,'p0+(q1+q2)*p1+sqrt(abs(q2))*p2',p0,p1,p2);
```

and plot it via

```
sareaplot(q1,q2,s)
```

What you get is a really nice picture

sarea, sareaplot



illustrating which combinations of parameter values yield a stable polynomial.

It is a must here to use names rather than values as the input arguments for both the parameters and the polynomials. Violation of this rule causes an error

```
S=sarea(-6:.1:12,q2,'p0+(q1+q2)*p1+sqrt(abs(q2))*p2',p0,p1,p2);
```

```
??? Error using ==> sarea
```

The input argument of parameter vector or polynomial must be a named variable.

Example 2

For the same three fixed polynomials, but a different uncertainty structure

$$p(s, \lambda_1, \lambda_2) = p_0(s) + \lambda_1 \lambda_2 p_1(s) + \lambda_2^3 p_2(s)$$

sarea, sareaplot-4

and parameters $\lambda_1 \in [-20, 20]$ and $\lambda_2 \in [-10, 10]$, we can use the gridding

```
lambda1=-20:.1:20;lambda2=-10:.1:10;
```

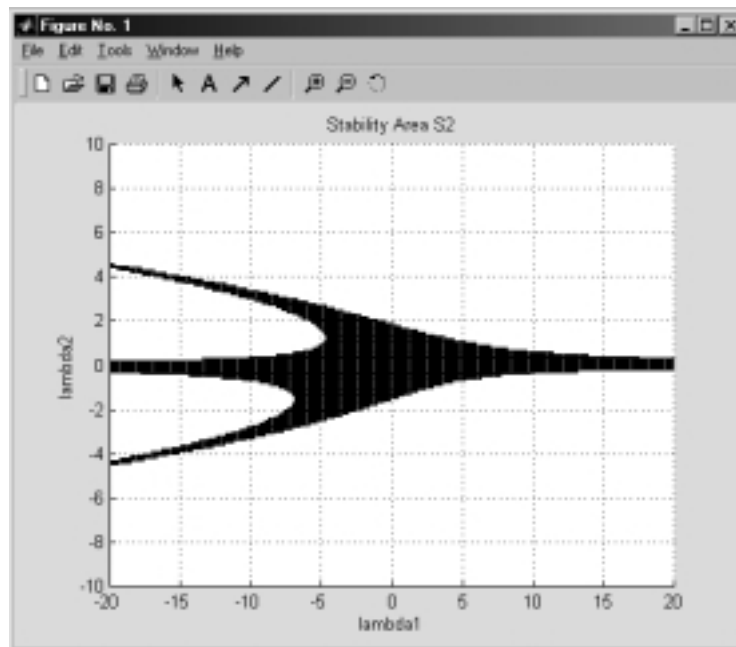
and type

```
expr='p0+lambda1*lambda2*p1+lambda2^2*p2';
```

```
S2=sarea(lambda1,lambda2,expr,p0,p1,p2);
```

```
sareaplot(lambda1,lambda2,S2)
```

to obtain another funny picture



Example 3

3-D examples are even nicer but, of course, more time consuming. Consider a three-parameter uncertain polynomial

$$p(s, q_1, q_2, q_3) = p_0(s) + (q_1 + q_1 q_3) q_3 p_1(s) + q_1^2 q_2^2 p_2(s)$$

with

sarea, sareaplot

$$p_0(s) = 2 + 4s + 3s^2 + s^3$$

$$p_1(s) = -1.7 + 0.13s + 0.29s^2$$

$$p_2(s) = 1.2 + 1.2s - 0.038s^2$$

and $q_1, q_2, q_3 \in [-20, 20]$. When inputting the data

```
p0=2+4*s+3*s^2+s^3;
```

```
p1=-1.7+0.13*s+0.29*s^2;
```

```
p2=1.2+1.2*s-0.038*s^2;
```

```
q1=-20:.5:20;q2=q1;q3=q1;
```

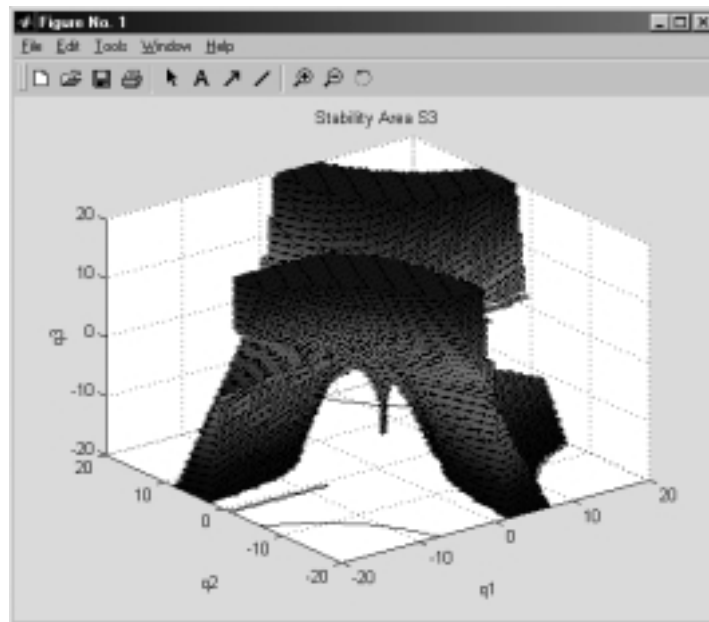
```
expr='p0+(q1+q1*q2)*q3*p1+(q1^2*q2^2)*p2';
```

the function called by

```
S3=sarea(q1,q2,q3,expr,p0,p1,p2);
```

needs more than one hour on an average PC to get the following result

```
sareaplot(q1,q2,q3,S3)
```



sarea, sareaplot-6

Such a 3-D plot can of course be zoomed or rotated by mouse in standard Matlab manner.

Example 4

For the same three fixed polynomials but another uncertainty structure

$$p(s, q_1, q_2, q_3) = p_0 + (q_1^2 - q_3)p_1 + (q_2 + q_3)p_2$$

with

$$q_1 \in [-7, 7], q_2 \in [-40, 2], q_3 \in [0, 40]$$

we enter the data

```
expr='p0+(q1^2-q3)*p1+(q2+q3)*p2';
```

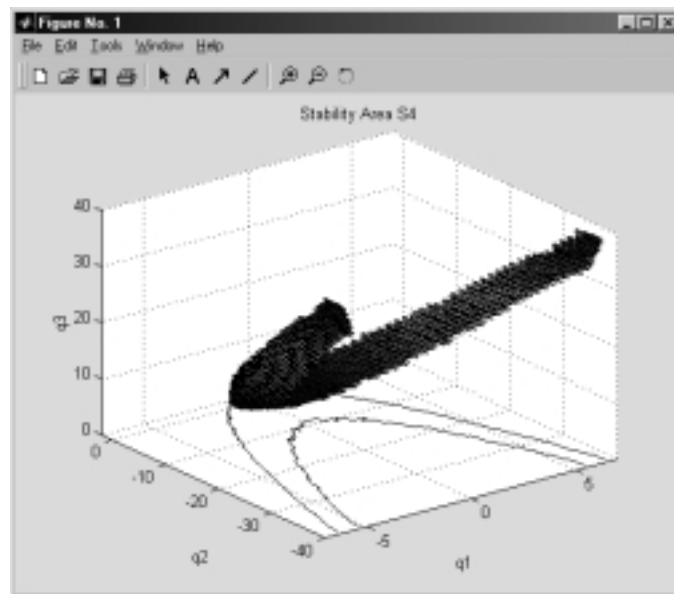
```
q1 = -7:.1:7; q2=-40:2; q3 = 0:0.5:40;
```

and run the macros

```
S4=sarea(q1,q2,q3,expr,p0,p1,p2);
```

```
sareaplot(q1,q2,q3,S4)
```

to obtain



sarea, sareaplot

Algorithm The method is trivial: It directly runs a stability test step by step for each particular point of the grid.

Diagnostics The macro `sarea` displays an error messages if

- There are not enough input arguments.
- An argument corresponding to parameter or polynomial is not a named variable.
- An invalid argument is encountered
- The expression string cannot be evaluated (in which case the error message is generated by `lasterr` and hence its text may vary according to the situation encountered).

The macro `sareaplot` displays an error messages if

- An invalid argument or option is encountered.
- There are more than three vectors representing uncertain parameters.
- Input arguments have inconsistent dimensions.

See also `isstable` stability test for a polynomial matrix
 `vset, vsetplot` value set plot for a parametric polynomial family