Contents

Demo file generation CRG_DEMO

Building a set of demo files with differ specifications. Do not alter this CRG-file. If necessary add new demo files. Several test proceedings require these data-structures. The file comments are optimized for the matlab publishing makro.

```
%   Copyright 2005-2011 OpenCRG - Daimler AG - Jochen Rauh
%
%   Licensed under the Apache License, Version 2.0 (the "License");
%   you may not use this file except in compliance with the License.
%   You may obtain a copy of the License at
%
%       http://www.apache.org/licenses/LICENSE-2.0
%
%   Unless required by applicable law or agreed to in writing, software
%   distributed under the License is distributed on an "AS IS" BASIS,
%   WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
%   See the License for the specific language governing permissions and
%   limitations under the License.
%
%   More Information on OpenCRG open file formats and tools can be found at
%
```

```
%       http://www.opencrg.org
%
%   $Id: crg_demo.m 184 2010-09-22 07:41:39Z jorauh $
```

Demo proceeding

The demos are initialized as followed:

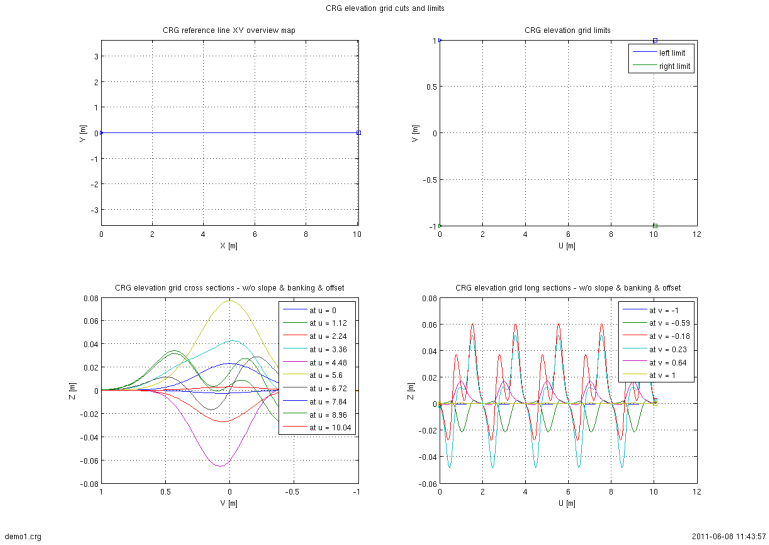
Demo 1-9

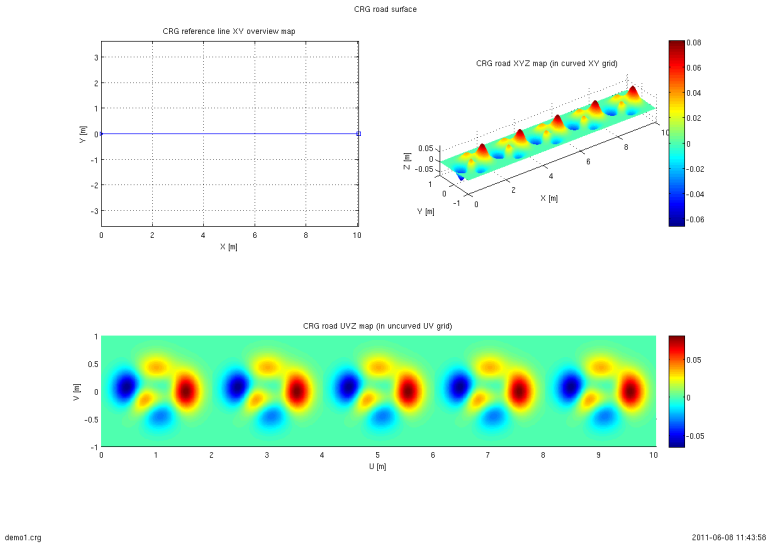
- generate minimal crg-structure
- alter, add specifications (change increment, add slope ec.)
- write CRG-file
- show results

```
% DEFAULT SETTINGS
% clear enviroment
clear all;
close all
% display results
dispRes = 1;

% build minimum crg-struct
uinc = 0.01;
vinc = 0.01;

nv = 201;
nu = 5*nv;

v = -(nv-1)/2*vinc:vinc:(nv-1)/2*vinc;

z = 0.01*peaks(nv);
z = repmat(z, nu/nv, 1);
```

Demo1: crg defined by z matrix and scalar u and v specs

```
data.u = (nu-1)*uinc;
data.v = (nv-1)*vinc/2;
data.z = z;
data.ct{1} = 'CRG defined by z matrix';
crg_write(crg_single(data), 'demo1.crg');

dat = crg_read('demo1.crg');
if dispRes, crg_show(dat); end
```





CRG elevation grid size:

length	=	[illegible]
width	=	[illegible]

CRG header data:

reference line start u	=	[illegible]
reference line end u	=	[illegible]
reference line increment	=	[illegible]
long section v right	=	[illegible]
long section v left	=	[illegible]
long section v increment	=	[illegible]
reference line start x	=	[illegible]
reference line end x	=	[illegible]
reference line offset x	=	[illegible]
reference line start y	=	[illegible]
reference line end y	=	[illegible]
reference line offset y	=	[illegible]
reference line start z	=	[illegible]
reference line end z	=	[illegible]
reference line offset z	=	[illegible]
reference line start phi	=	[illegible]
reference line end phi	=	[illegible]
reference line offset phi	=	[illegible]
reference line start s	=	[illegible]
reference line end s	=	[illegible]
reference line start b	=	[illegible]
reference line end b	=	[illegible]

CRG modifier data:

grid nan mode	=	[illegible]
refpoint u fraction	=	[illegible]
refpoint u offset	=	[illegible]
refpoint v	=	[illegible]
refpoint v offset	=	[illegible]
refpoint x	=	[illegible]
refpoint y	=	[illegible]
refpoint z	=	[illegible]
refpoint phi	=	[illegible]

CRG option data:

border mode u	=	[illegible]
border mode v	=	[illegible]
border offset u	=	[illegible]
border offset v	=	[illegible]
border smooth ubeg	=	[illegible]
border smooth uend	=	[illegible]
refline continuation	=	[illegible]
refline search far	=	[illegible]
refline search close	=	[illegible]
check eps	=	[illegible]
check inc	=	[illegible]
check tol	=	[illegible]

CRG derived data:

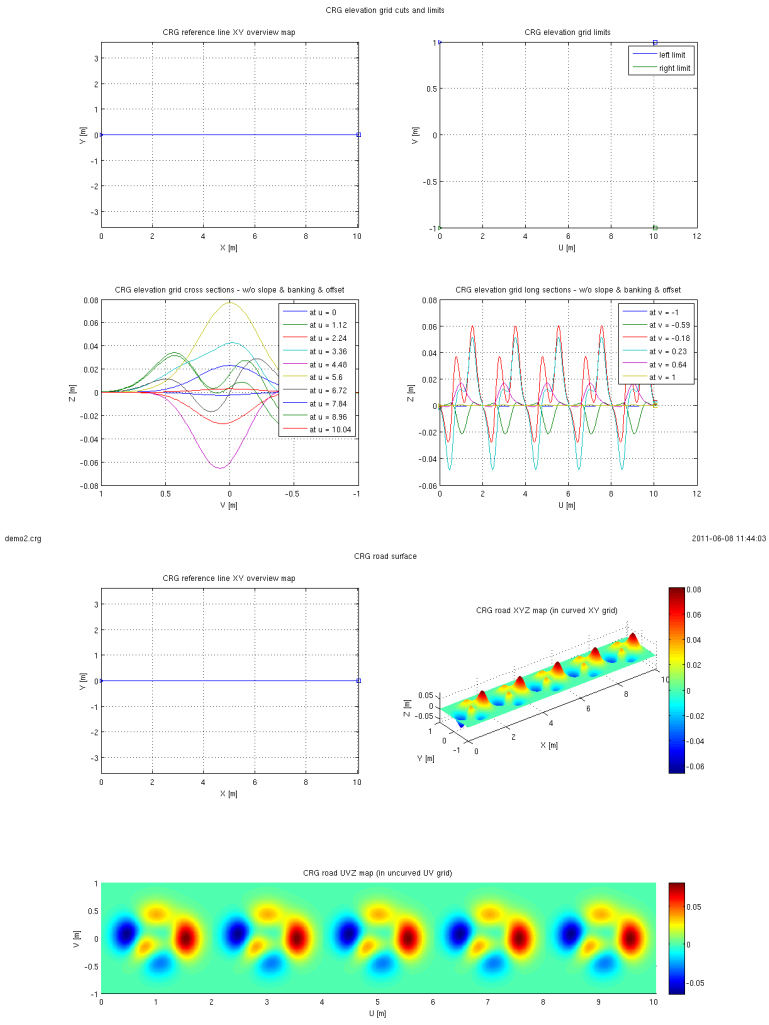
crossing ubeg	=	[illegible]
crossing uend	=	[illegible]
crossing ulen	=	[illegible]

CRG comment data:

CRG defined by z matrix

Demo2: ... and evenly spaced v vector

```
data.v = v;
data.ct{2} = '... and evenly spaced v vector';
crg_write(crg_single(data), 'demo2.crg');

dat = crg_read('demo2.crg');
if dispRes, crg_show(dat); end
```



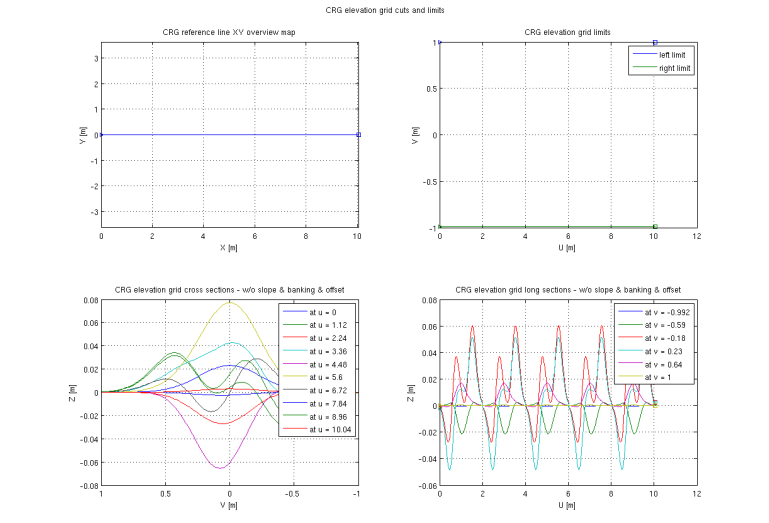
```
CRG elevation grid size:
length                    =    [illegible]
width                     =    [illegible]

CRG header data:
reference line start u    =    [illegible]
reference line end u      =    [illegible]
reference line increment  =    [illegible]
long section v right      =    [illegible]
long section v left       =    [illegible]
long section v increment  =    [illegible]
reference line start x    =    [illegible]
reference line end x      =    [illegible]
reference line offset x   =    [illegible]
reference line start y    =    [illegible]
reference line end y      =    [illegible]
reference line offset y   =    [illegible]
reference line start s    =    [illegible]
reference line end s      =    [illegible]
reference line offset s   =    [illegible]
reference line start phi  =    [illegible]
reference line end phi    =    [illegible]
reference line offset phi =    [illegible]
reference line start z    =    [illegible]
reference line end z      =    [illegible]
reference line start b    =    [illegible]
reference line end b      =    [illegible]

CRG modifier data:
grid nan mode             =    [illegible]
refpoint u fraction       =    [illegible]
refpoint u offset         =    [illegible]
refpoint v                =    [illegible]
refpoint v offset         =    [illegible]
refpoint x                =    [illegible]
refpoint y                =    [illegible]
refpoint s                =    [illegible]
refpoint phi              =    [illegible]

CRG option data:
border mode u             =    [illegible]
border mode v             =    [illegible]
border offset u           =    [illegible]
border offset v           =    [illegible]
border smooth ubeg        =    [illegible]
border smooth uend        =    [illegible]
refline continuation      =    [illegible]
refline search far        =    [illegible]
refline search close      =    [illegible]
check eps                 =    [illegible]
check inc                 =    [illegible]
check tol                 =    [illegible]

CRG derived data:
crossing ubeg             =    [illegible]
crossing uend             =    [illegible]
crossing ulen             =    [illegible]
```

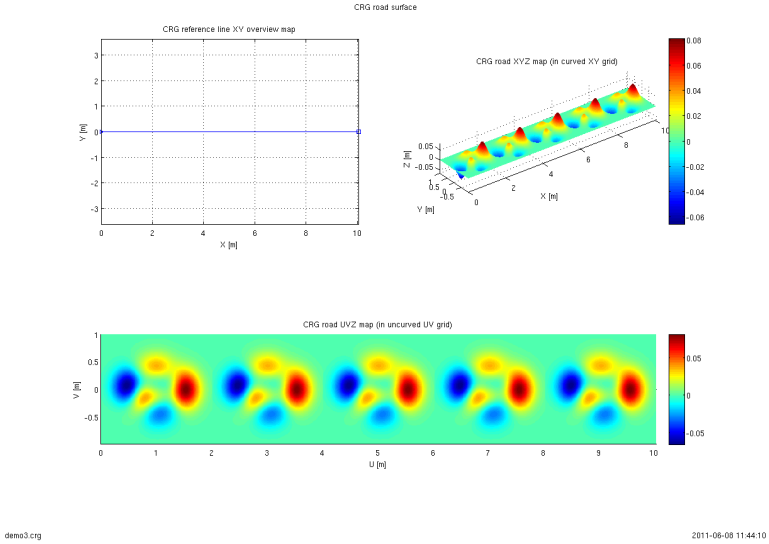
```
CRG comment data:

CRG defined by z matrix
... and evenly spaced v vector
```

demo2.crg 2011-06-08 11:44:05

Demo3: ... and unevenly spaced v vector

```
data.v(1) = single(-0.992);
data.ct{2} = '... and unevenly spaced v vector';
crg_write(crg_single(data), 'demo3.crg');

dat = crg_read('demo3.crg');
if dispRes, crg_show(dat); end
```





CRG elevation grid size:

length	=	[illegible]
width	=	[illegible]

CRG header data:

reference line start u	=	[illegible]
reference line end u	=	[illegible]
reference line increment	=	[illegible]
long section v right	=	[illegible]
long section v left	=	[illegible]
reference line start x	=	[illegible]
reference line end x	=	[illegible]
reference line offset x	=	[illegible]
reference line start y	=	[illegible]
reference line end y	=	[illegible]
reference line offset y	=	[illegible]
reference line start s	=	[illegible]
reference line end s	=	[illegible]
reference line offset s	=	[illegible]
reference line start phi	=	[illegible]
reference line end phi	=	[illegible]
reference line offset phi	=	[illegible]
reference line start z	=	[illegible]
reference line end z	=	[illegible]
reference line start b	=	[illegible]
reference line end b	=	[illegible]

CRG modifier data:

grid nan mode	=	[illegible]
refpoint u fraction	=	[illegible]
refpoint u offset	=	[illegible]
refpoint v	=	[illegible]
refpoint v offset	=	[illegible]
refpoint x	=	[illegible]
refpoint y	=	[illegible]
refpoint z	=	[illegible]
refpoint phi	=	[illegible]

CRG option data:

border mode u	=	[illegible]
border mode v	=	[illegible]
border offset u	=	[illegible]
border offset v	=	[illegible]
border smooth ubeg	=	[illegible]
border smooth uend	=	[illegible]
refline continuation	=	[illegible]
refline search far	=	[illegible]
refline search close	=	[illegible]
check eps	=	[illegible]
check inc	=	[illegible]
check tol	=	[illegible]

CRG derived data:

crossing ubeg	=	[illegible]
crossing uend	=	[illegible]
crossing ulen	=	[illegible]

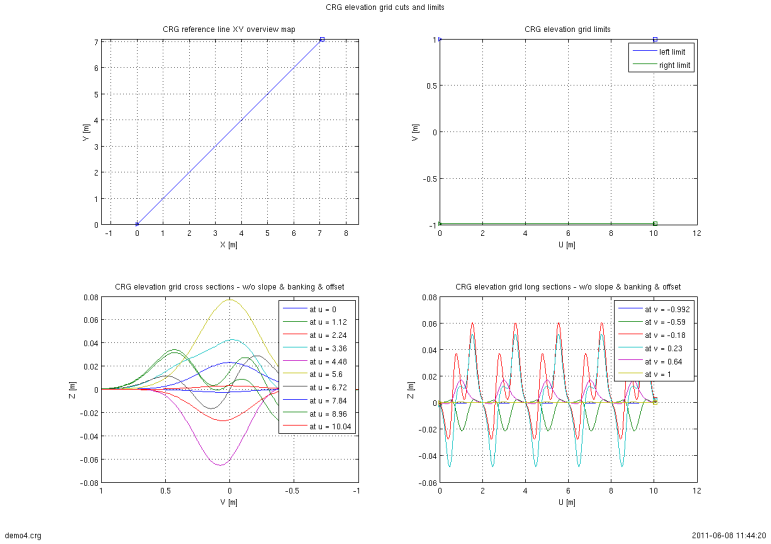
CRG comment data:

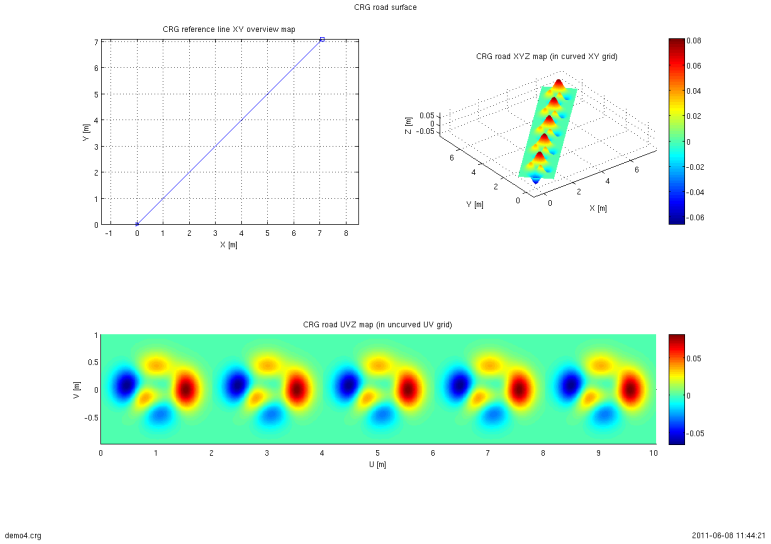
CRG defined by z matrix
... and unevenly spaced v vector

demo3.crg 2011-06-08 11:44:17

Demo4: ... generate diagonal reference line by one p value

```
data.p(1) = pi/4;
data.ct{3} = '... with diagonal reference line by one p value';
crg_write(crg_single(data), 'demo4.crg');

dat = crg_read('demo4.crg');
if dispRes, crg_show(dat); end
```





CRG elevation grid size:

length	=	[illegible]
width	=	[illegible]

CRG header data:

reference line start u	=	[illegible]
reference line end u	=	[illegible]
reference line increment	=	[illegible]
long section v right	=	[illegible]
long section v left	=	[illegible]
reference line start x	=	[illegible]
reference line end x	=	[illegible]
reference line offset x	=	[illegible]
reference line start y	=	[illegible]
reference line end y	=	[illegible]
reference line offset y	=	[illegible]
reference line start z	=	[illegible]
reference line end z	=	[illegible]
reference line offset z	=	[illegible]
reference line start phi	=	[illegible]
reference line end phi	=	[illegible]
reference line offset phi	=	[illegible]
reference line start s	=	[illegible]
reference line end s	=	[illegible]
reference line start b	=	[illegible]
reference line end b	=	[illegible]

CRG modifier data:

grid nan mode	=	[illegible]
refpoint u fraction	=	[illegible]
refpoint u offset	=	[illegible]
refpoint v	=	[illegible]
refpoint v offset	=	[illegible]
refpoint x	=	[illegible]
refpoint y	=	[illegible]
refpoint z	=	[illegible]
refpoint phi	=	[illegible]

CRG option data:

border mode u	=	[illegible]
border mode v	=	[illegible]
border offset u	=	[illegible]
border offset v	=	[illegible]
border smooth ubeg	=	[illegible]
border smooth uend	=	[illegible]
refline continuation	=	[illegible]
refline search far	=	[illegible]
refline search close	=	[illegible]
check eps	=	[illegible]
check inc	=	[illegible]
check tol	=	[illegible]

CRG derived data:

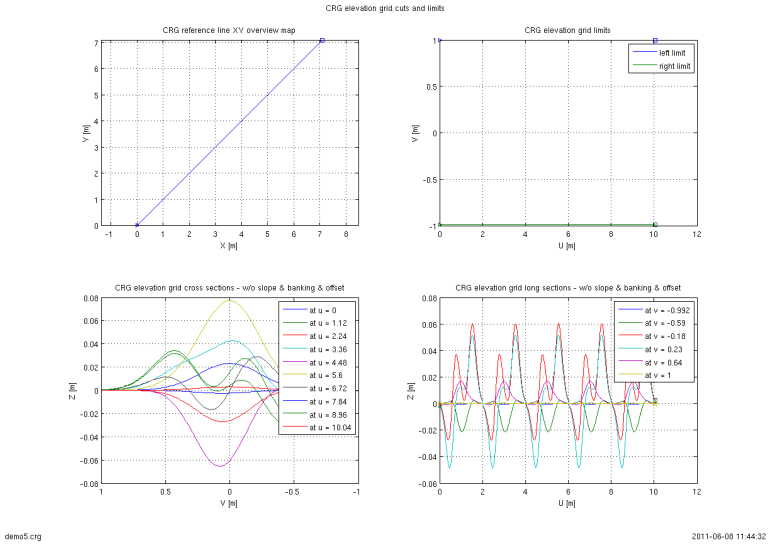
crossing ubeg	=	[illegible]
crossing uend	=	[illegible]
crossing uinc	=	[illegible]

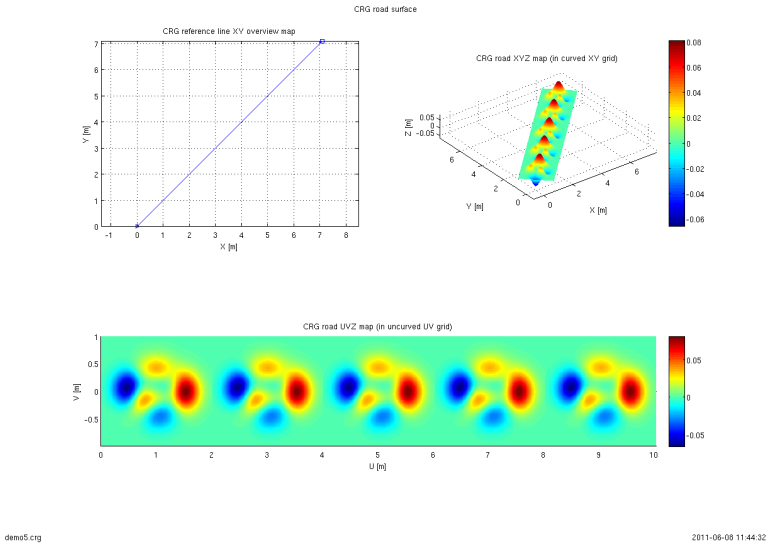
CRG comment data:

CRG defined by z matrix
... and unevenly spaced v vector
... with diagonal reference line by one p value

Demo5: ... generate diagonal reference line by nu-1 p values

```
np = nu-1;
data.p(1:np) = pi/4;
data.ct{3} = '... with diagonal reference line by nu-1 p values';
crg_write(crg_single(data), 'demo5.crg');

dat = crg_read('demo5.crg');
if dispRes, crg_show(dat); end
```





CRG elevation grid size:

length	=	[illegible]
width	=	[illegible]

CRG header data:

reference line start u	=	0
reference line end u	=	[illegible]
reference line increment	=	[illegible]
long section v right	=	[illegible]
long section v left	=	[illegible]
reference line start x	=	0
reference line end x	=	[illegible]
reference line offset x	=	0
reference line start y	=	0
reference line end y	=	[illegible]
reference line offset y	=	0
reference line start z	=	0
reference line end z	=	0
reference line offset z	=	0
reference line start phi	=	[illegible]
reference line end phi	=	[illegible]
reference line offset phi	=	0
reference line start s	=	0
reference line end s	=	0
reference line start b	=	0
reference line end b	=	0

CRG modifier data:

grid nan mode	=	[illegible]
refpoint u fraction	=	0
refpoint u offset	=	0
refpoint v	=	0
refpoint v offset	=	0
refpoint x	=	0
refpoint y	=	0
refpoint z	=	0
refpoint phi	=	0

CRG option data:

border mode u	=	[illegible]
border mode v	=	[illegible]
border offset u	=	0
border offset v	=	0
border smooth ubeg	=	0
border smooth uend	=	0
refline continuation	=	0
refline search far	=	[illegible]
refline search close	=	[illegible]
check eps	=	[illegible]
check inc	=	[illegible]
check tol	=	[illegible]

CRG derived data:

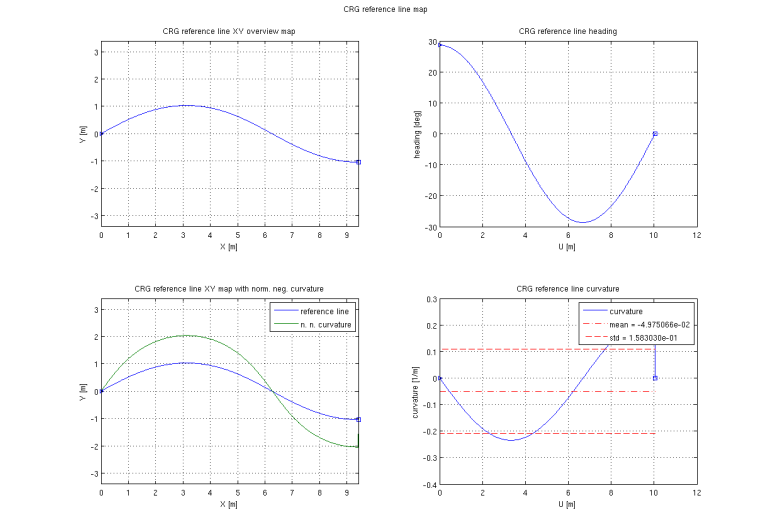
crossing ubeg	=	0
crossing uend	=	[illegible]
crossing ulen	=	0

CRG comment data:

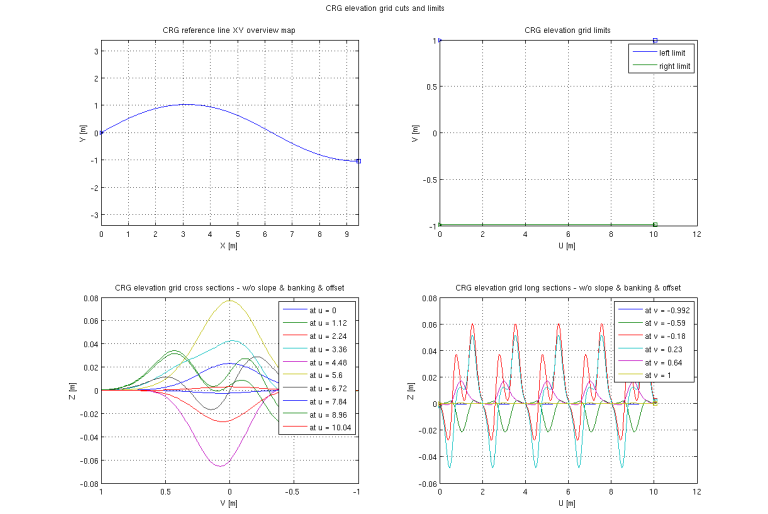
CRG defined by z matrix
... and unevenly spaced v vector
... with diagonal reference line by nu-1 p values

Demo6: ... generate curved reference line

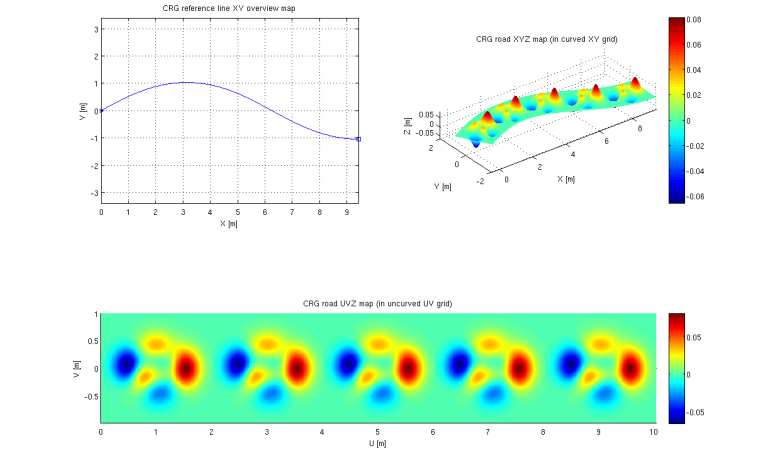
```
np = nu-1;
for i=1:np
    data.p(i) = 0.5*cos(i/np*1.5*pi);
end

data.ct{3} = '... with curved reference line';
crg_write(crg_single(data), 'demo6.crg');

dat = crg_read('demo6.crg');
if dispRes, crg_show(dat); end
```



demo6.crg 2011-06-08 11:44:43



demo6.crg 2011-06-08 11:44:44



demo6.crg 2011-06-08 11:44:45

CRG information

CRG elevation grid size:

length	=	[illegible]
width	=	[illegible]

CRG header data:

reference line start u	=	[illegible]
reference line end u	=	[illegible]
reference line increment	=	[illegible]
long section v right	=	[illegible]
long section v left	=	[illegible]
reference line start x	=	[illegible]
reference line end x	=	[illegible]
reference line offset x	=	[illegible]
reference line start y	=	[illegible]
reference line end y	=	[illegible]
reference line offset y	=	[illegible]
reference line start z	=	[illegible]
reference line end z	=	[illegible]
reference line offset z	=	[illegible]
reference line start phi	=	[illegible]
reference line end phi	=	[illegible]
reference line offset phi	=	[illegible]
reference line start s	=	[illegible]
reference line end s	=	[illegible]
reference line start b	=	[illegible]
reference line end b	=	[illegible]

CRG modifier data:

grid nan mode	=	[illegible]
refpoint u fraction	=	[illegible]
refpoint u offset	=	[illegible]
refpoint v	=	[illegible]
refpoint v offset	=	[illegible]
refpoint x	=	[illegible]
refpoint y	=	[illegible]
refpoint z	=	[illegible]
refpoint phi	=	[illegible]

CRG option data:

border mode u	=	[illegible]
border mode v	=	[illegible]
border offset u	=	[illegible]
border offset v	=	[illegible]
border smooth ubeg	=	[illegible]
border smooth uend	=	[illegible]
refline continuation	=	[illegible]
refline search far	=	[illegible]
refline search close	=	[illegible]
check eps	=	[illegible]
check inc	=	[illegible]
check tol	=	[illegible]

CRG derived data:

crossing ubeg	=	[illegible]
crossing uend	=	[illegible]
crossing ulen	=	[illegible]

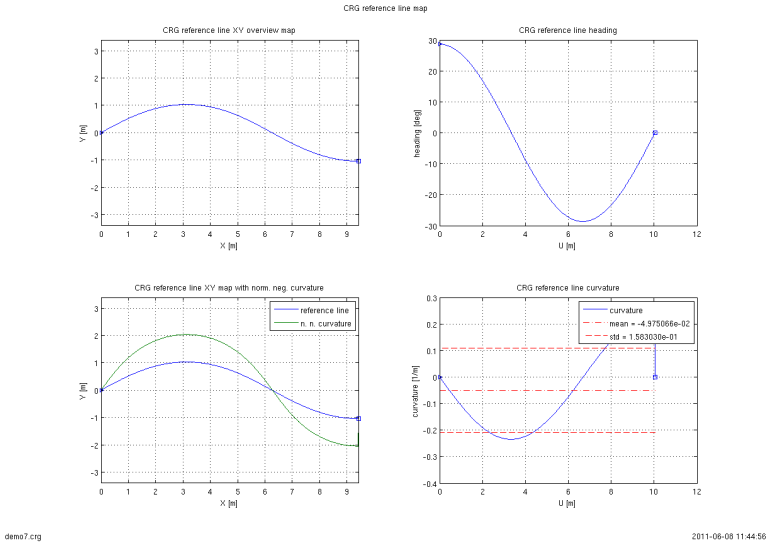
CRG comment data:

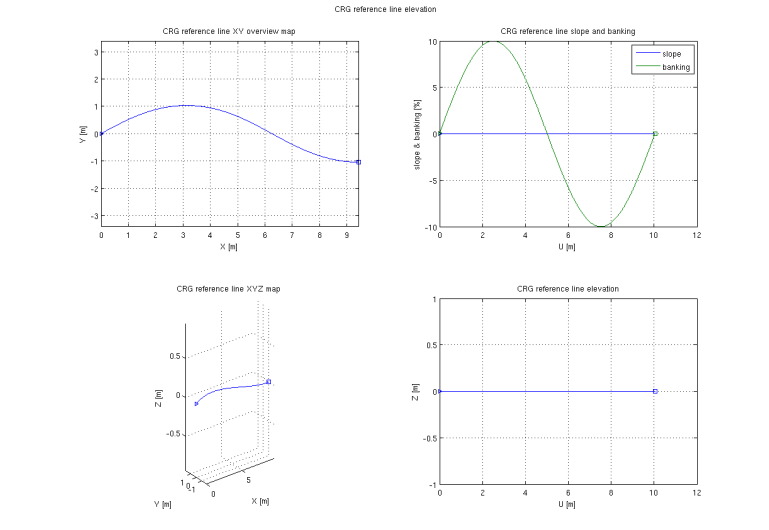
CRG defined by z matrix
... and unevenly spaced v vector
... with curved reference line

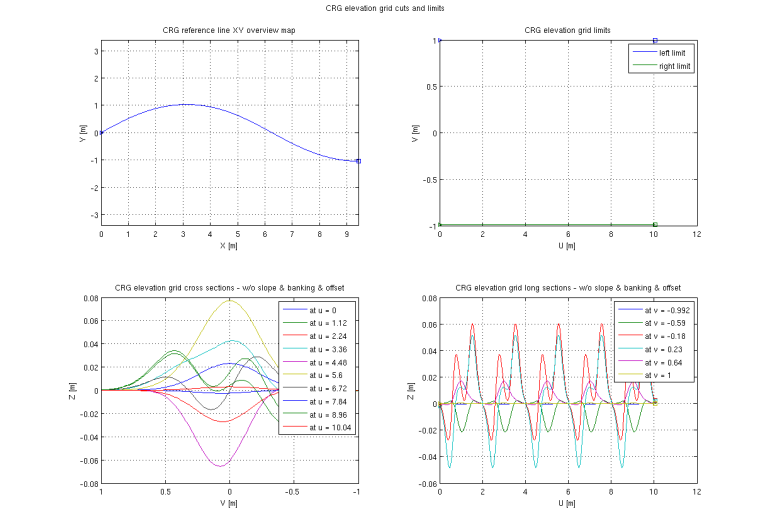
demo6.crg 2011-06-08 11:44:52

Demo7: ... generate banking

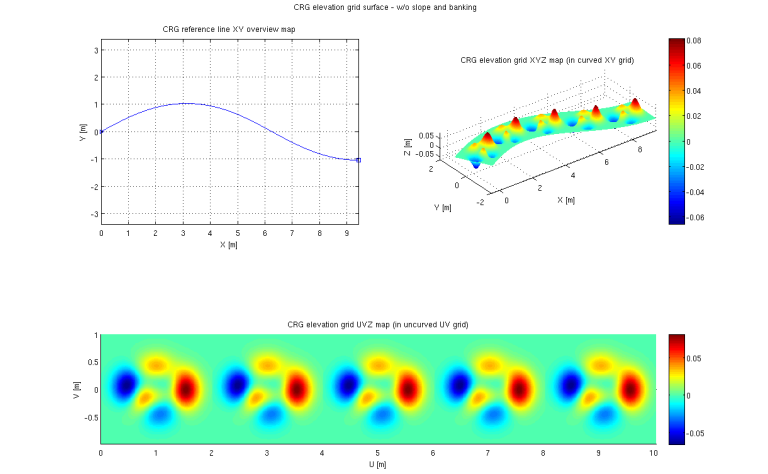
```
for i=1:nu
    data.b(i) = 0.1*sin(i/nu*2*pi);
end
data.ct{4} = '... with variable cross slope';
crg_write(crg_single(data), 'demo7.crg');

dat = crg_read('demo7.crg');
if dispRes, crg_show(dat); end
```





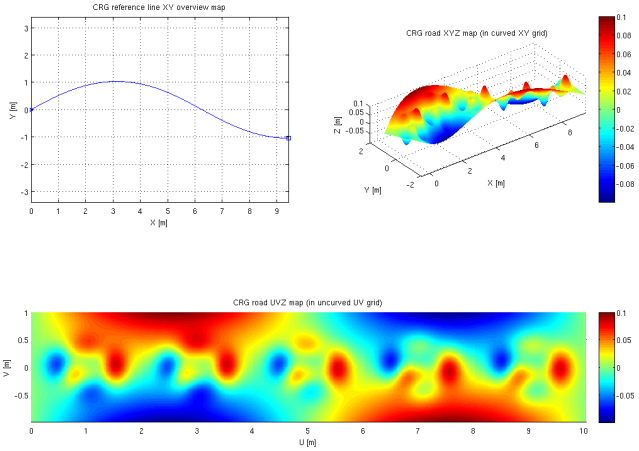


demo7.crg 2011-06-08 11:44:58



demo7.crg 2011-06-08 11:44:58

CRG road surface



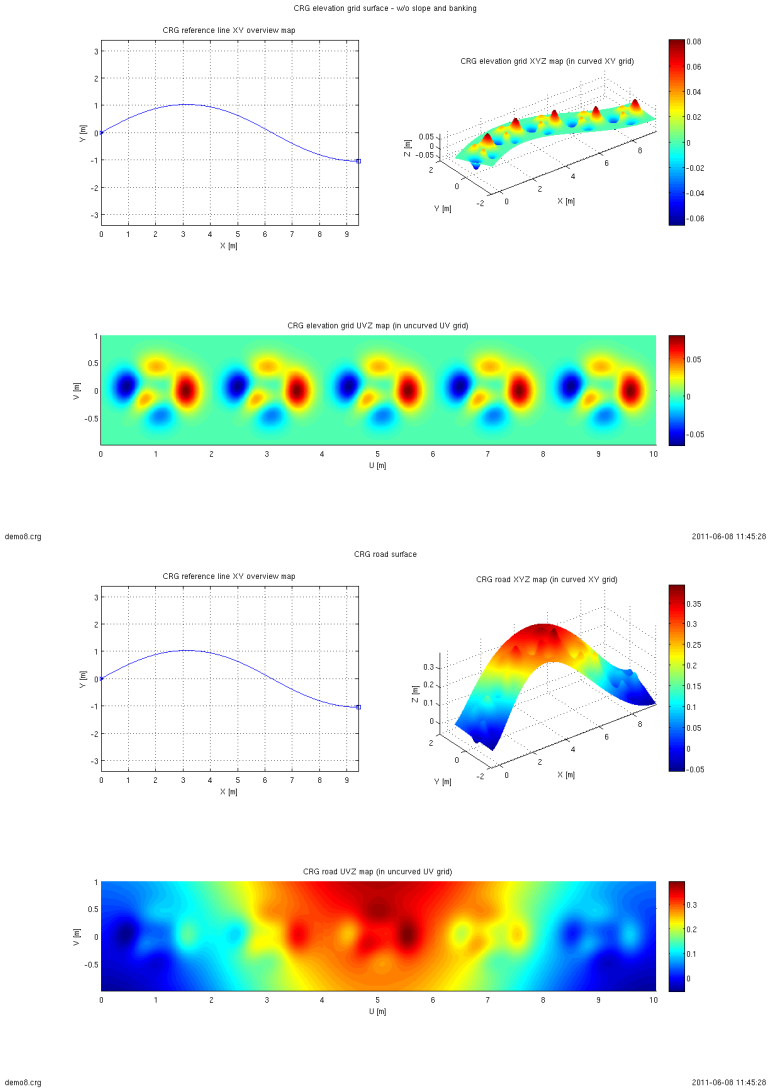
CRG information

CRG comment data:

CRG defined by Z matrix
... and unevenly spaced v vector
... with curved reference line
... with variable cross slope

Demo8: ... generate slope

```
for i=1:np
    data.s(i) = 0.1*sin(i/np*2*pi);
end
data.b=0.05; % constant cross slope
data.ct{4} = '... with variable slope and constant cross slope';
crg_write(crg_single(data), 'demo8.crg');

dat = crg_read('demo8.crg');
if dispRes, crg_show(dat); end
```



CRG information

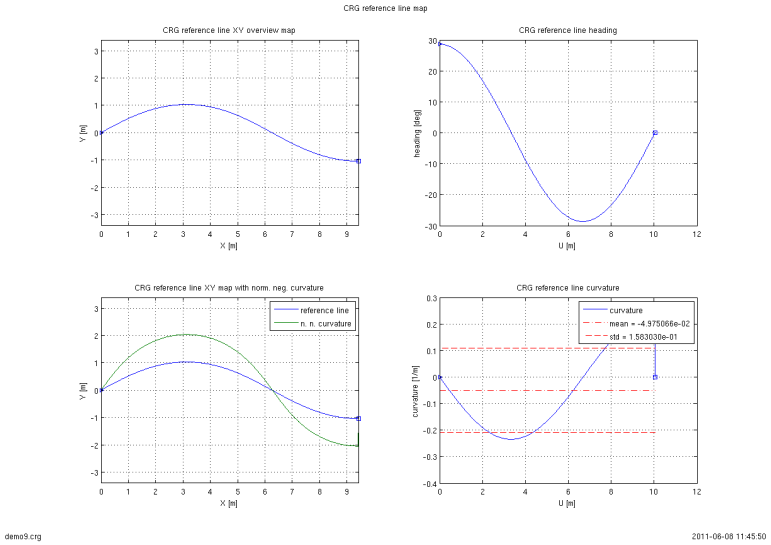
CRG elevation grid size:		
length	=	3115
width	=	213
CRG header data:		
reference line start u	=	0
reference line end u	=	31.14
reference line increment	=	0.01
long section v right	=	0.992
long section v left	=	0
reference line start x	=	0
reference line end x	=	[illegible]
reference line offset x	=	0
reference line start y	=	0
reference line end y	=	[illegible]
reference line offset y	=	0
reference line start z	=	0
reference line end z	=	[illegible]
reference line offset z	=	0
reference line start phi	=	[illegible]
reference line end phi	=	[illegible]
reference line offset phi	=	0
reference line start s	=	[illegible]
reference line end s	=	[illegible]
reference line start b	=	[illegible]
reference line end b	=	[illegible]
CRG modifier data:		
grid nan mode	=	2
refpoint u fraction	=	0
refpoint u offset	=	0
refpoint v	=	0
refpoint v offset	=	0
refpoint x	=	0
refpoint y	=	0
refpoint z	=	0
refpoint phi	=	0
CRG option data:		
border mode u	=	2
border mode v	=	2
border offset u	=	0
border offset v	=	0
border smooth ubeg	=	0
border smooth uend	=	0
refline continuation	=	0
refline search far	=	1.5
refline search close	=	0.3
check eps	=	1e-06
check inc	=	0.001
check tol	=	0.0001
CRG derived data:		
crossing ubeg	=	0
crossing uend	=	31.14
crossing ulen	=	0

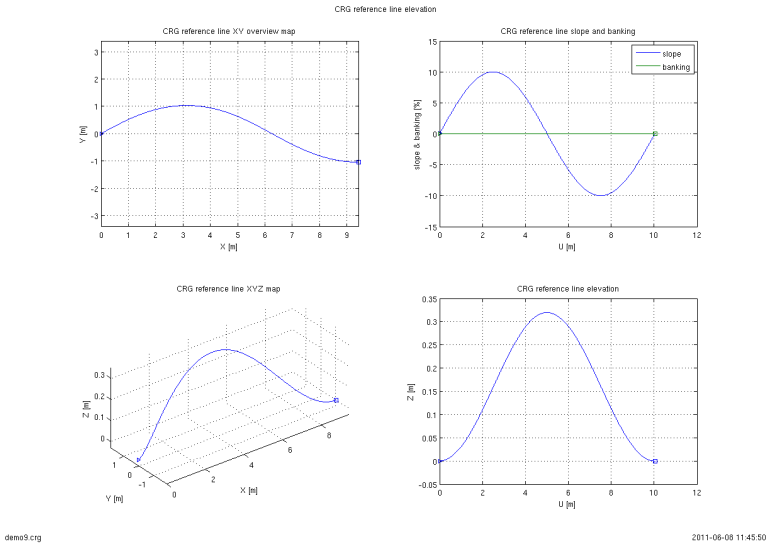
```
CRG comment data:

CRG defined by z matrix
... and unevenly spaced v vector
... with curved reference line
... with variable slope and constant cross slope
```

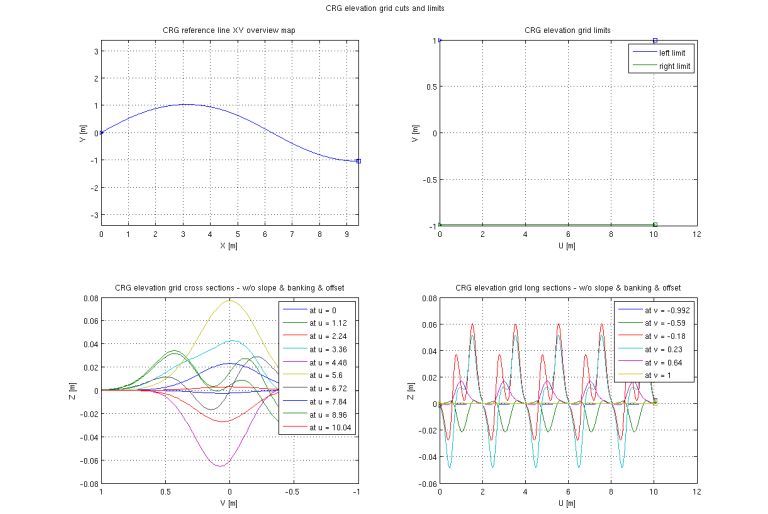
demo8.crg 2011-06-08 11:45:45

Demo9: ... slope without banking

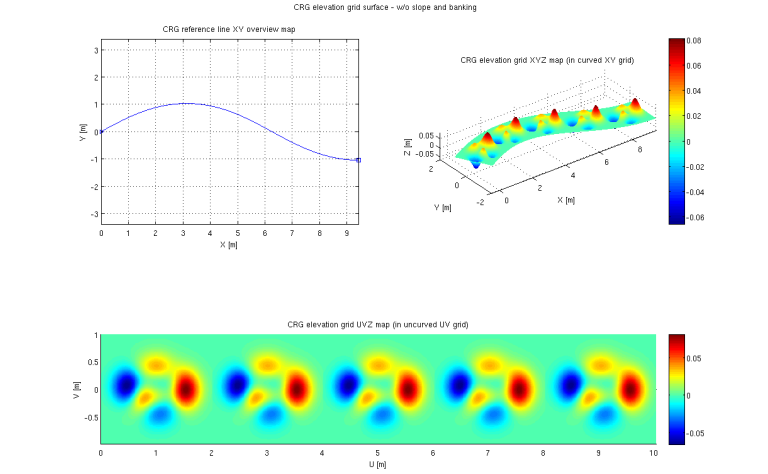
```
data = rmfield(data, 'b');

data.ct{4} = '... with variable slope';
crg_write(crg_single(data), 'demo9.crg');

dat = crg_read('demo9.crg');
if dispRes, crg_show(dat); end
```



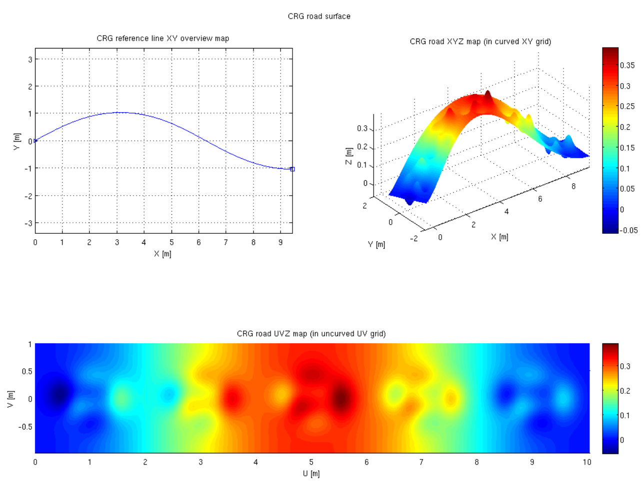




demo9.crg 2011-06-08 11:45:51



demo9.crg 2011-06-08 11:45:52



demo9.crg 2011-06-08 11:45:53

CRG information

```
CRG elevation grid size:
length                       =   [illegible]
width                        =   [illegible]

CRG header data:
reference line start u       =   [illegible]
reference line end u         =   [illegible]
reference line increment     =   [illegible]
long section v right         =   [illegible]
long section v left          =   [illegible]
reference line start x       =   [illegible]
reference line end x         =   [illegible]
reference line offset x      =   [illegible]
reference line start y       =   [illegible]
reference line end y         =   [illegible]
reference line offset y      =   [illegible]
reference line start s       =   [illegible]
reference line end s         =   [illegible]
reference line offset s      =   [illegible]
reference line start phi     =   [illegible]
reference line end phi       =   [illegible]
reference line offset phi    =   [illegible]
reference line start z       =   [illegible]
reference line end z         =   [illegible]
reference line start b       =   [illegible]
reference line end b         =   [illegible]

CRG modifier data:
grid nan mode                =   [illegible]
refpoint u fraction          =   [illegible]
refpoint u offset            =   [illegible]
refpoint v                   =   [illegible]
refpoint v offset            =   [illegible]
refpoint x                   =   [illegible]
refpoint y                   =   [illegible]
refpoint z                   =   [illegible]
refpoint phi                 =   [illegible]

CRG option data:
border mode u                =   [illegible]
border mode v                =   [illegible]
border offset u              =   [illegible]
border offset v              =   [illegible]
border smooth ubeg           =   [illegible]
border smooth uend           =   [illegible]
refline continuation         =   [illegible]
refline search far           =   [illegible]
refline search close         =   [illegible]
check eps                    =   [illegible]
check inc                    =   [illegible]
check tol                    =   [illegible]

CRG derived data:
crossing ubeg                =   [illegible]
crossing uend                =   [illegible]
crossing ulen                =   [illegible]
```

```
CRG comment data:

CRG defined by z matrix
... and unevenly spaced v vector
... with curved reference line
... with variable slope
```

demo9.crg 2011-06-08 11:46:09