

## README

```
#-----  
#                               | SLIPREAL |  
#-----  
#  
# A collection of MATLAB programs to generate stochastic slip  
# distributions, in particular useful for calculating near-source  
# strong ground motions from a "realistic" faulting model.  
#  
# -----  
# Author: P. Martin Mai (mai@seismo.ifg.ethz.ch)  
# Institute of Geophysics, ETH Hoenggerberg  
# CH-8093 Zurich, Switzerland  
# http://seismo.ethz.ch/staff/martin/martin.html  
# -----  
#  
# REQUIREMENTS  
#  
# In order to run the m-files of the SLIPREAL-package, you need  
# to have (at least) MATLAB 5.0 with the following toolboxes:  
# signal-processing and statistics. Aside from that it should  
# run on any platform supporting MATLAB.  
#  
# INSTALLATION  
#  
# Unzip the SlipReal.zip file into a directory of your choice,  
# and add this directory to your MATLAB search path. That will  
# make sure you can run SlipReal whenever you launch MATLAB.  
#  
# RUNNING SLIPREAL  
#  
# There is a (very small) example (Example1.mat) that you can  
# use to get started, containing sample input structures and  
# output arrays generated by SlipReal. Aside from that, this  
# README or typing 'help SlipReal' at the MATLAB prompt is all  
# you need in order to successfully generate stochastic slip  
# distributions.  
#  
# DISCLAIMER  
#  
# These programs (m-files) come at no guarantee whatsoever, and  
# I am not responsible for the results other people obtain and  
# publish based on the use of these codes. SLIPREAL has been  
# tested in various applications, but certainly not exhaustively,  
# and I do expect that users find bugs and errors. Please report  
# them directly to me, and I try to fix them.  
# Using these routines may not appear very straight-forward,  
# user-friendly or well-documented -- that is simply because the  
# package has grown over time, and originally was not meant to  
# be distributed. I may go back to rewrite part of it sometime  
# in the far future to enhance user-friendliness. In any case,  
# whenever you have questions or comments, please contact me;  
# I'll try to be of as much help as possible, but I certainly  
# cannot run the programs for you.  
#  
# REQUEST  
#  
# If you find these programs useful, and you get to the point of
```

```
# publishing work in which you used SLIPREAL, please reference  
# the following two publications. Thanks.  
#  
# Mai, P.M., and G.C. Beroza (2002). A spatial random-field model  
# to characterize complexity in earthquake slip, J. Geoph. Res.,  
# 107(B11), 2308, doi:10.1029/2001JB000588, 2002.  
# Mai, P.M., and G.C. Beroza (2000). Source-scaling properties  
# from finite-fault rupture models, Bull. Seis. Soc. Am., 90,  
# 604-615.  
#  
#  
# HAVE FUN !!  
#  
# Martin Mai, May 2003  
# -----  
#  
# THE PROGRAMS  
#  
# SlipReal is using the work of Mai & Beroza (2002) in which we  
# show that finite-source slip models of past earthquakes can be  
# characterized by a spatial random-field model. Testing various  
# auto-correlation functions, we observe that the correlation  
# lengths scale with seismic moment. Assuming a fractal model,  
# inferred finite-source models show a slightly different fractal  
# dimension as proposed by Andrews (1980). These results, in  
# conjunction with scaling relations on the source dimensions  
# (Mai & Beroza, 2000; Wells & Coppersmith, 1994), can be used  
# to generate stochastic slip distributions which capture the  
# slip complexity as imaged for past earthquakes.  
#  
# Having provided that code with the essential fault parameters,  
# fault width, fault length, moment magnitude and faulting style,  
# the code lets you choose the auto-correlation function, the  
# corresponding correlation lengths (fractal dimension), spatial  
# sampling etc, in order to generate a stochastic slip map.  
# The underlying algorithm is based on the spectral synthesis,  
# described by Pardo-Iguzquiza and Chica-Olma (1993).  
#  
# TO GET STARTED  
#  
# You can get a head-start by just giving the source parameters  
# fault width, length, and moment magnitude as srcpar = [W L M],  
# and the faulting mechanism FM as 'ss' or 'ds' for strike-slip  
# or dip-slip earthquakes, respectively. E.g., to simulate a  
# slip on the fault plane for an earthquake like the 1995 Kobe  
# (Japan) earthquake (length ~60km, width ~20km, Mw = 6.9), just  
# set srcpar = [20 60 6.9]; and then call SlipReal as follows  
# [S,par]=SlipReal(srcpar,'ss')  
# The code will run once, telling you which default values it has  
# chosen, displaying the resulting slip map, and generating the  
# output structure PAR that can then be used to run the code with  
# different input values. As simple as that! Of course you have  
# more control over the results if you specify the entire set of  
# input arguments (as given below).  
#  
# NOTE  
#  
# In case you do want to compute synthetics seismograms for the
```

```
% slip model(s) you generate, I have put together a few small
% routines to simulate the hypocentral position, which can then
% be checked against the slip map (RandHypo.m, CheckHypo.m).
% The ratio of rupture- to shear-wave velocity can also be
% randomized (RandVrat.m), and, provided you specify a velocity-
% density structure (see rpar in Example1.mat), be used to
% compute the rupture onset times over the fault plane (function
% RupTimeGrid2.m). Together with CalcTrfromM.m, which computes
% a single, constant rise-time value for the entire fault, the
% SlipReal package could be used to generate a complete
% KINEMATIC source model for strong motion calculations.
% All that remains is to define an appropriate slip-velocity
% function, and deploy a code to compute the synthetics near-
% source seismograms (i.e. discrete-wavenumber, isochrone,
% finite-differences).
% CAUTION: This source characterization is kinematic, and hence
% generates source models that may not be physically realizable.
% Nothing in the code prevents you to have the rupture propagate
% at 99% light speed, with rise time 1e8 secs!. So double-check
% your results with some rupture dynamic considerations in the
% back of your mind. We are working on putting more earthquake
% physisc in these type of simulations (for references see
% http://seismo.ethz.ch/staff/martin/publications.html), and
% plan to publish/distribute that as soon as we are confident
% in our methodology.
%
% Below you find (hopefully) all you need to understand and run
% SLIPREAL. You get the same information, if you type
% 'help SlipReal' at the MATLAB prompt.
%
%-----
%
% [S,par] = SlipReal(srcpar,'mech','acf',corr,seed,samp,grd, ...
%                   nexpt,wlevel,taper,depth,dip,outfile,fig);
%
% simulates a dislocation model for given source parameters and
% source mechanism. If not given, source dimensions can be
% computed from empirical scaling relations.
% The slip on the fault surface is calculated using the spectral
% synthesis method in which the slip-amplitude spectrum is defined
% through a spatial auto-correlation function or a power law decay;
% the phase is random in U(-pi,pi), but the entire field obeys
% Hermitian symmetry.
% The variables SRCPAR and MECH are required input; for all other
% input parameters, an empty array [] will select the default values
% (see below).
%
% INPUT:
% srcpar - array-structure with ALL source parameters OR
%          cell-array with source parameters and string to identify scaling;
%          OR simple vector with source parameters
%          for the last two options, srcpar can be of the form
%          {Mw 'rel'} -- source dimensions computed from scaling laws
%                   rel == 'MB' uses Mai & Beroza (2000)
%                   rel == 'WC' uses Wells & Coppersmith (1994)
%                   rel == 'WG' uses USGS WorkingGroup99 M vs. A (2000)
%          {A 'rel'} -- area A (in km^2), rel is 'MB' or 'WC'
%          [W L] -- Mw estimated from Wells & Coppersmith (1994),
%          [W L Mw] -- D computed from given parameters
%          [W L Mw D] -- Mw will be scaled to match given D!
%
% NOTE: give fault width W and length L in km, mean slip D in m
%
% If srcpar is given as array-structure, make sure the naming
```

```
%
% in your array is EXACT the same as needed in the code;
% the best idea is to run SlipReal once with standard
% input values, and then modify the entries in the
% output-structure PAR which then can be used as input.
%
% mech - faulting mechanism for the simulated event, needed to
%         compute source parameters from scaling relations
%         'ss' or 'SS' for strike-slipevents;
%         'ds' or 'DS' for dip-slip events
%         'al' or 'AL' if both types should be considered
%         (may be useful in case of normal faulting events)
%
% acf - string to denote autocorrelation function
%       'ak' or 'AK' for anisotropic von Karman ACF
%       'ex' or 'EX' for exponential ACF
%       'fr' or 'FR' for the fractal case (power law decay)
%       'gs' or 'GS' for Gaussian ACF [needs input corr. length]
%       if [], default 'ak' is used
%
% corr - correlation length and/or spectral decay parameters
%        [az ax] for Gauss or exponential ACF
%        [az ax H] for von Karman ACF H = Hurst number)
%        [D kc] for fractal slip where D is the fractal dimension;
%             kc: corner wavenumber beyond which the spectrum decays;
%             kc is related to the source dimensions, and is computed
%             as  $kc = 2\pi/(\sqrt{L*W})$  if it is not given
%        [] if corr is an empty matrix, the relevant parameters for
%             the given autocorrelation function will be computed
%             (NOT true for the Gaussian since no relations have been
%             established between corr length and source parameters)
%
% seed - structural array of seed values for the random-number generator,
%        called at various locations in the code; if seed == [], then
%        the code uses an expression like
%             'Rseed = sum(100*clock);'
%             'randn('seed', Rseed);' <-- uses MATLAB4 generators!!
%        to generate the random numbers; the value Rseed is stored in
%        the output structure par.
%        The sequence is as follows, and is the same that is returned
%        by the code:
%             seed.SS = SSseed; 1x2-array, used in SpecSyn2
%             seed.WL = WLseed; 1x1-array, used in WaterLevel
%             seed.CS = CSseed; 2x2-array, used in CalcSigfromD
%             seed.RC = RCseed; 3x2-array, used in CalcDimfromM
%             seed.RWC = RWCseed; 3x2-array, used in CalcDimWC
%             seed.CR = CRseed; 1x1, 4x2, 5x2, used in CalcCorrfromLW
%
%        Hence, you can run the code once, get the corresponding array
%        and use it again to create the EXACT SAME random slip model.
%
% samp - sampling of dislocation model in z,x direction [dz dx]
%        NOTE: the final sampling may be finer in case the option 'pow2'
%             is given as 'y' (see SPECSYN2) or sampling must
%             be adjusted for the source dimensions
%
% grd - slip function to be defined on grid-nodes or subfaults
%       'nod' for grid-node definition [grid-size (L/dx+1)*(W/dz+1)]
%       'sub' for sub-fault definition [grid-size (L/dx) * (W/dz)]
%
% nexpt - non-linear scaling exponent for the random field (i.e.  $S = S^{nexpt}$ )
%         nexpt < 1 smoothenes the fields (steepens the spectrum)
%         nexpt == 1 doesn't change anything (default)
%         nexpt > 1 roughens the field (flattens the spectrum)
%         the purpose of this variable is to allow for simulation of
```

```
%      slip distributions with large peak-slip values;
%      a good choice for that is usually nexpt = 1.25 - 1.5;
%
% wlevel - method to scale the zero-mean random field to nonnegative slip
%          wlevel == []: field "lifted" above zero (default)
%          wlevel == 0: values < 0 will be set equal to zero
%          wlevel == -1: values < 0 will be set to small random value
%          wlevel == -2: values < 0 will be set to 0.25*field value
%          wlevel == -3: values < mean-slip set to small random value
%          wlevel == scalar: values < scalar set to small random value
%          NOTE: wlevel = [] preserves the spectral decay; wlevel = -2
%          also preserves the slip spectrum rather well without intro-
%          ducing too much artificial high-wavenumbers, yet scales the
%          slip such that locally large-slip patches are present
%
% taper - tapering the edges of the slip model
%          [left/right top bottom] (in km) for customized tapering
%          'y' for default tapering of 2.5 km (i.e [2.5 2.5 2.5])
%          [left/right top bottom P] where P is an exponent smaller 1,
%          applies an additional 'depth-taper' of the form  $z^P$ 
%          (i.e. some kind of sqrt-function) to mimick less slip in the
%          upper crustal regions close to the surface
%
% depth - max. depth of rupture plane;
%          option with depth range [zmin zmax] not implemented
%          zmin is the depth to the top of the fault plane
%          zmax is the maximum depth of fault plane (in km)
%          (default: zmin = 0, zmax =15)
%
% dip - dip angle of fault plane (in degrees) (default = 90 deg)
%
% fig - optional: 'y' to view the slip realization; this will open
%          a figure window for each realization (set to 'n' if called
%          in a loop) (default: 'y')
%
% outfile- optional: string for a filename to which the code writes
%          the slip realization as a simple ascii 2D-array, where rows
%          correspond to depth (default: 'n')
%
% OUTPUT:
% S - 2D-slip distribution
% par - structure with all relevant source parameters
#
# -----
#
#          END OF README FOR SLIPREAL
#
# -----
```