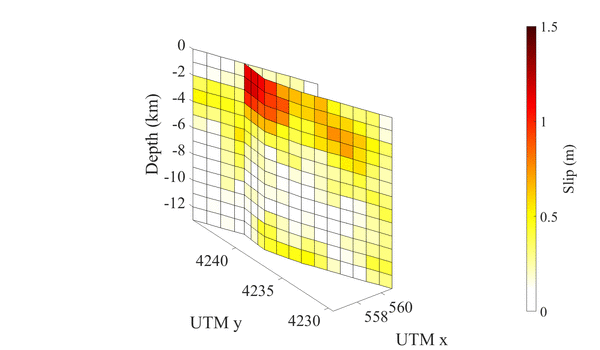
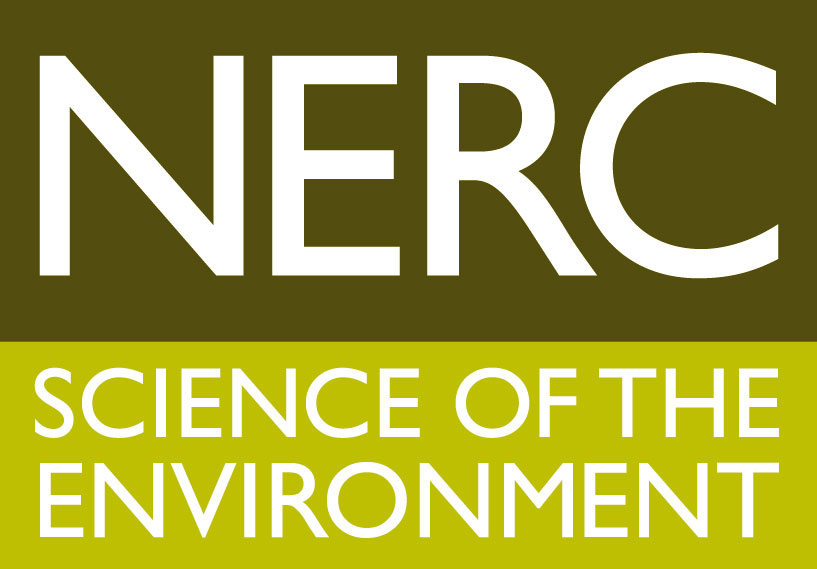
slipBERI

- **slip** **B**ay**E**sian **R**egularised **I**nversion -



Author: R.M.J. Amey, Jan 2019



## **Contents**

[Contents](#_jslmnz6jud9o)

[Introduction](#_lswhqql3em27)

[Installation](#_ixjjtvkp2u7n)

[Data Preparation](#_c3rsonf5lmx2)

[GPS](#_bcw6mbsbpyzp)

[InSAR](#_r8izx5knsp0b)

[Fault](#_xe14uja9arc)

[Input file](#_gi5ubpx71g46)

[Fault](#_n7xz6lllqz7p)

[Data](#_la8nw1ok98fx)

[Invert](#_mizdfr720fqs)

[Priors](#_e01fs2ee0qwa)

[Elastic\_params](#_uhv4qxwcfr6)

[Display](#_oh63gr8uqk6t)

[Housekeeping](#_fsn3t2uypmfh)

[Data errors](#_qin3hpag8d3u)

[Running the inversion](#_97z8e3yyk0xz)

[Start the inversion](#_hd49pfq2p4e6)

[Initial plots and data check](#_5ehu6vlodmpm)

[During the inversion](#_ih1bwp9p6yq6)

[At the end of the inversion](#_ik0jpfok25)

[References](#_bkv7fcq9ewh1)

## 

## 

## **Introduction**

The open-source code slipBERI allows the user to invert geodetic data (Interferometric Synthetic Aperture Radar (InSAR) or Global Navigation Satellite System (GNSS)) to derive a slip distribution on a given fault. The primary purpose of this function is to incorporate observed properties of fractal earthquake slip (e.g. Sagy 2007, Candela 2011, Milliner 2015) into the slip inversion as a form of regularisation (after Hooper 2012). This is done through the von Karman autocorrelation function (Mai & Beroza, 2002).

The function inverts for slip in a Bayesian fashion, using a Markov chain Monte Carlo (MCMC) algorithm to search parameter space, incorporating the Metropolis-Hastings algorithm (Metropolis 1953, Hastings 1970). For more information on Bayesian methods I direct the reader to Mosegaard and Tarantola 1995, Tarantola 2005, Fukuda and Johnson 2008. The fault is a dislocation embedded in an elastic half-space (Okada 1985).

slipBERI is written in *Matlab* and uses many sub-functions to run. It is made available for non-commercial applications only and can be downloaded here: <https://github.com/ruthamey/slipBERI>

**For more information and when using this software please reference Amey et al., 2018:**

Amey, R. M. J., Hooper, A., & Walters, R. J. (2018). A Bayesian method for incorporating self‐similarity into earthquake slip inversions. *Journal of Geophysical Research: Solid Earth*, 123, 6052–6071. <https://doi.org/10.1029/2017JB015316>

Or, if using the transdimensional approach:

Amey, R. M. J., Hooper, A., & Morishita, Y. ( 2019). Going to any lengths: Solving for fault size and fractal slip for the 2016, *Mw* 6.2 Central Tottori earthquake, Japan, using a transdimensional inversion scheme. *Journal of Geophysical Research: Solid Earth*, 124.<https://doi.org/10.1029/2018JB016434>

Through this manual, commands to be entered on the Matlab command line are in blue and preceded by the symbol >>. Screen outputs are in grey. Specific portions of the command that require modification by the user are in red.

Typing ‘help’ followed by the name of the function (for example >> help slipBERI) provides a brief description of the function and of the input/output parameters.

A very useful resource that can be used alongside this software is GBIS, the Geodetic Bayesian Inversion Software (Bagnardi and Hooper, 2018). It can be used in conjunction with slipBERI, to perform an initial inversion to determine the fault geometry for a uniform slip model, upon which you can then do a distributed slip inversion using slipBERI. This help document is based upon GBIS.

GBIS can be downloaded here: <https://comet.nerc.ac.uk/gbis/>

Bagnardi M. & Hooper A, (2018). Inversion of surface deformation data for rapid estimates of source parameters and uncertainties: A Bayesian approach. Geochemistry, Geophysics, Geosystems, 19. https://doi.org/10.1029/2018GC007585

**Important: some functions in slipBERI share a name with those in GBIS, but require slightly different inputs. So please make sure that you remove GBIS from your path before running slipBERI, or you may get strange error messages.**

## **Installation**

slipBERI is available to download here:

<https://github.com/ruthamey/slipBERI>

On the right, press the green ‘Clone or download’ button and either download a zip file or clone the file.

Move the zip file to an appropriate folder and unzip the contents, giving a slipBERI-master file. You should find the following directories and files:

/slipBERImbin

slipBERI.m

make\_structures\_required\_for\_slipBERI.m

README.txt

Open Matlab and add the slipBERI-master/ directory and subdirectories to your path:

>> addpath(genpath(‘path/to/your/local/directory/slipBERI-master/’))

Test the paths using the following script:

>> test\_paths

This will display

/yourpath/slipBERI-master/slipBERI.m

/yourpath/slipBERI-master/slipBERImbin/error\_messages.m

/yourpath/slipBERI-master/slipBERImbin/calc\_logprior\_VK.m

/yourpath/slipBERI-master/slipBERImbin/plotting/doplot3d.m

slipBERI and its functions are all set up

You are now ready to use slipBERI with your own data.

## **Data Preparation**

### **GPS**

slipBERI requires that GPS data is provided in a single text file (e.g. GPS\_data.txt). The file must have 7 columns separated by a space, and each GPS site will have three rows associated with it, for the separate East, North and Up components. The seven columns are:

1. Longitude in decimal degrees, positive if East and negative if West

or

UTM x value (km)

1. Latitude in decimal degrees, positive if North and negative if South.

or

UTM y value (km)

1. Displacement in meters
2. ‘1’ or ‘0’ or ‘0’. If the row relates to east-west displacement, then column 4 will have a 1. If it relates to displacement in a different displacement, it will have a 0.
3. ‘1’ or ‘0’ or ‘0’. If the row relates to north-south displacement, then column 5 will have a 1. If it relates to displacement in a different displacement, it will have a 0.
4. ‘1’ or ‘0’ or ‘0’. If the row relates to vertical displacement, then column 6 will have a 1. If it relates to displacement in a different displacement, it will have a 0.
5. Estimated error in corresponding direction (east-west, north-south or vertical) in meters, normally expressed as one standard deviation.

Rows 4,5 and 6 are the GPS ‘look vectors’. So where for InSAR data the look vector is comprised of east, north and up components, for GPS the look vector is purely east, purely north or purely vertical.

Example for two GPS stations:

237.2530 38.4407 -0.00546 1 0 0 0.00337

237.2530 38.4407 0.00161 0 1 0 0.00406

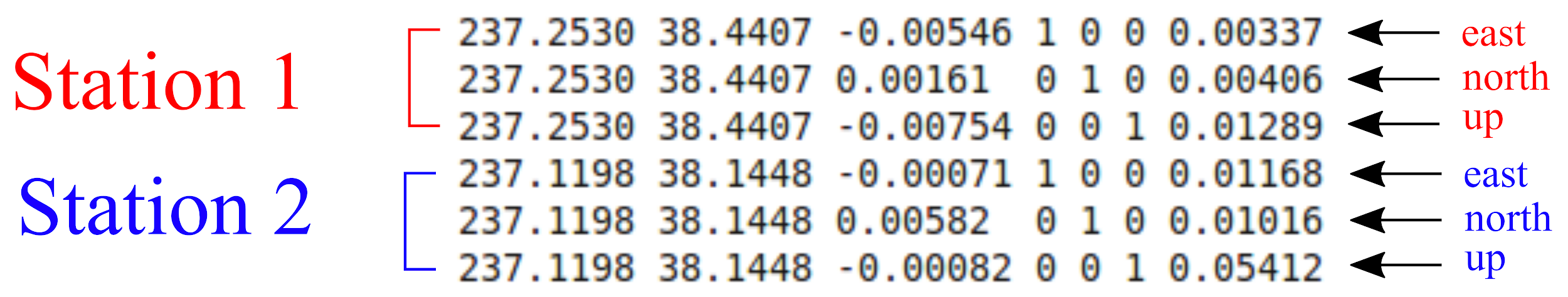
237.2530 38.4407 -0.00754 0 0 1 0.01289

237.1198 38.1448 -0.00071 1 0 0 0.01168

237.1198 38.1448 0.00582 0 1 0 0.01016

237.1198 38.1448 -0.00082 0 0 1 0.05412

where:



Please note that the GPS coordinate unit must be stated as appropriate in the ‘data’ structure.

data.GPS\_coordinate\_unit = ‘utm’

or

data.GPS\_coordinate\_unit = ‘long/ lat’

### **InSAR**

slipBERI requires that InSAR data is provided as text files, one for each InSAR dataset. The files must have 6 columns separated by a space. The six columns are:

1. Longitude in decimal degrees, positive if East and negative if West

or

UTM x value (km)

1. Latitude in decimal degrees, positive if North and negative if South.

or

UTM y value (km)

1. Line-of-sight displacement **in meters.** Very important: displacement is positive when towards the satellite (e.g. uplift) and negative away from the satellite (e.g. subsidence).
2. Line-of-sight vector east-west component. An east pointing vector would be positive, a west pointing vector would be negative. Very important: line of sight is positive looking **at** the satellite.
3. Line-of-sight vector north-south component. An north pointing vector would be positive, a south pointing vector would be negative.Very important: line of sight is positive looking **at** the satellite.
4. Line-of-sight vector up component. An up pointing vector would be positive, a down pointing vector would be negative.Very important: line of sight is positive looking **at** the satellite, so the up component of the line-of-sight vector should always be positive.

Please downsample your data prior to the inversion. As a rule of thumb, aim to have ~1000 datapoints across all the InSAR and GPS scenes, though more of fewer points may be required depending on the data.

To do quadtree downsampling [GBIS](https://comet.nerc.ac.uk/gbis/) or [kite](https://github.com/pyrocko/kite) are both nice pieces of software.

Or use a nested uniform downsampling approach.

Please note that the InSAR coordinate unit must be stated as appropriate in the ‘data’ structure.

data.InSAR\_coordinate\_unit = ‘utm’

or

data.InSAR\_coordinate\_unit = ‘long/ lat’

### Fault

The fault information must be presented in a text file of one row per fault strand and 11 columns, separated by a space. These columns are:

* Strike (degrees)
* Dip (degrees) -- using the Aki and Richards convention for strike and dip
* Rake (degrees) -- if solving for rake within the inversion this is just an initial value
* X center -- the up-dip location of the x center of the fault plane (either longitude or UTM x (m))
* Y center -- he up-dip location of the y center of the fault plane (either latitude or UTM y (m))
* Length (km)
* Top depth (km) -- Where down is positive. If the fault breaks the surface this will be 0km, if buried it will be 5km
* Bottom depth -- Down is positive
* Patches along strike -- users choice of how many patches to solve for
* Patches down dip -- users choice of how many patches to solve for
* Smoothing identifier -- fault strands with the same smoothing identifier will be smoothed across, if invert.smooth\_across\_fault\_strands = ‘yes’. This allows the user to input several fault strands within the fault descriptor file in order to capture the curving geometry of a fault, but still smooth across them. So if one fault curved and so was expressed as three different fault strands, plus there was another separate fault not to smooth over, then the first three fault strands would have ‘1’ in this column, and the next row would have ‘2’.
* 1 -- this last column relates to a function that is not fully operational and so must be ‘1’.

Please note that the fault coordinate unit must be stated as appropriate in the ‘fault’ structure.

fault.fault\_coordinate\_unit = ‘utm’

or

fault.fault\_coordinate\_unit = ‘long/ lat’

## Input file

The function slipBERI requires seven structures in order to run. These structures are set up in the script make\_structures\_required\_for\_slipBERI.m

These structures are:

* fault - contains details about the fault model upon which slip will be inverted
* data - contains details about the data which will be used in the inversion
* invert - contains details of how the inversion will be performed
* priors - contains details about the prior information which will be used in the inversion
* elastic\_params - values for Lame’s first and second parameters
* display - contains details of which parameters to plot at the end
* housekeeping - choose a savename for the inversion

In this section each of the values in these structures are explained. These can be changed by editing the script, and then once the correct details have been inputted, in the matlab terminal run as follows:

>> make\_structures\_required\_for\_slipBERI.m

### **Fault**

'Fault' is a structure relating to the description of the fault model upon which distributed slip will be inverted.

fault.fault\_descriptor\_file = ‘name\_of\_file\_with\_fault\_details.txt’

String. Name of text file with details of fault geometry, as described above.

fault.fault\_coordinate\_unit = ‘long/lat’

String. 'utm' or 'long/lat' to indicate whether the coordinates used in fault.fault\_desriptor\_file are in given in UTM or long/lat.

### **Data**

'Data' is a structure giving details of the data provided for the inversion and how they should be weighted.

data.InSAR\_datafile = ‘name\_of\_InSAR\_datafile.txt’

String. Name of text file with details of InSAR data, details of format given above. If you have no InSAR data is being used this MUST be 'none'.

data.InSAR\_coordinate\_unit = ‘long/lat’

'utm' or 'long/lat'. Everything will be converted to UTM or a local coordinate system as necessary.

data.varcovar\_details = ‘text\_file\_with\_sill\_nugget\_range.txt’

String. this is a text file with the [sill, variogram\_range] from the covariogram from an undeforming region, details given below. **Important:** the variance-covariance details text files must be in data.varcovar\_details in the same order as the InSAR datafiles are listed in data.InSAR\_datefile

data.quadtree\_n\_points = ‘text\_file\_with\_details\_of\_how\_many\_points\_are\_averaged\_in\_each\_pixel.txt’

String. Name of a text file with the number of pixels combined into each datapoint, if using quadtree. This is used to calculate the var-covar matrix. If not using quadtree this is a text file with '1's in each row, the same length as the InSAR datafile. **Important:** the quadtree\_n\_points details text files must be in data.quadtree\_n\_points in the same order as the InSAR datafiles are listed in data.InSAR\_datefile

data.GPS\_datafile\_2d = ‘name\_of\_GPS\_2d\_file.txt’

String. Name of GPS file of 2d surface displacement, details of format given above. If no 2D GPS data is being used this MUST be 'none'. Locations translated to UTMx or a local latitude, if spanning multiple UTM zones.

data.GPS\_datafile\_3d = ‘name\_of\_GPS\_3d\_file.txt’

String. Name of GPS file of 3d surface displacement. If no 3D GPS data is being used this MUST be 'none'. Locations translated to UTMx or

data.GPS\_coordinate\_unit = ‘long/lat'

String. 'lat/long' or 'utm'. Everything will be converted to UTM or a local coordinate system as necessary.

data.atolls\_datafile = ‘none’

String. Name text file with uplift only data in.

data.atolls\_coordinate\_unit = ‘long/lat'

String. 'utm' or 'long/lat'

data.weight\_InSAR = 1

Number. Relative weighting of InSAR data. If not weighting datasets then this must be 1. If this is a number other than 1 then the InSAR var-covar matrix will be divided by this value.

data.weight\_GPS = 1

Number. Relative weighting of GPS data. If not weighting datasets then this must be 1. If this is a number other than 1 then the GPS var-covar matrix will be divided by this value.

data.weight\_atolls = 1

Number. Relative weighting of atolls data. If not weighting datasets then this must be 1. If this is a number other than 1 then the atolls var-covar matrix will be divided by this value.

data.UTMzone = UTM\_zone

Number (ignoring letter). The UTM zone e.g. 10. Negative if in southern hemisphere e.g. -10.

data.origin = [origin\_long, origin\_lat]

A long/lat in the centre of your area [-124 38]. This is only used if your data spans multiple UTM zones and in that case everything is converted to a local coordinate system, using the origin.

data.EQ\_epicenter = [EQ\_lat, EQ\_long, EQ\_depth].

Number. Matrix of [EQ\_lat, EQ\_long, EQ\_depth]. Depth in km. Used for plotting only.

data.seismic\_moment = moment\_value

Number, in Newton Meters. Only used if regularising the inversion using the moment. The USGS is a good source for a seismic moment estimate.

data.moment\_std = moment\_std\_value

Number, in Newton Meters. Only used if regularising the inversion using the moment. The USGS is a good source for a seismic moment estimate, and if there are multiple estimates then the standard deviation can be derived.

### **Invert**

Contains details on how the inversion will be performed i.e. which parameters will be solved for.

invert.quickcheck = ‘yes’

String. 'yes' or 'no'. Before commencing the inversion, your data will be plotted for you to click on to say that it's okay.

invert.inversion\_type = ‘bayesian’

String. 'least\_squares' or 'bayesian'.

invert.iterations = 100000

Number. Number of iterations, only relevant for Bayesian inversions.

invert.smoothing = ‘VK’

String. Either ‘VK’ for von Karman smoothing, ‘laplacian’ for Laplacian smoothing or ‘none’ for an unsmoothed inversion.

invert.smooth\_across\_fault\_strands = ‘yes’

String. 'yes' smoothes across different fault strands as if it were one fault. 'no' treats each fault as a separate slipping areas. This is only relevant if more than one fault is being used (i.e. the fault text file as named in fault.fault\_descriptor\_file would have more than one row).

invert.slip\_initial = 0.5

Number. Estimate of slip values in meters. All slip patches will be assigned this amount of slip initially, with a bit of noise added. This choice of number should not affect the final result, but choosing well may reduce the burn-in time.

invert.step\_size = 0.01

Value. Maximum step size, plus or minus, initially allowed in your random walk when generating a slip\_trial, for Bayesian inversions. This is used initially and then is adjusted through the iteration. The units are the same as the model parameter to which the step is being added.

invert.solve\_for\_fault\_size = ‘yes’

String. 'yes' or 'no'. **Important:** this only works for one fault strand currently. More than one fault or a series of connected faults are not currently supported.

invert.variable\_rake = ‘yes’

String. 'yes' or 'no'

invert.solve\_for\_InSAR\_offset = ‘no’

String. 'yes' or 'no' for whether to solve for an offset between each InSAR scene and the data.

invert.solve\_for\_InSAR\_ramp = ‘no’

String. 'yes' or 'no' for whether to solve for a ramp and InSAR offset for each InSAR scene and the data. Note that if invert.solve\_for\_InSAR\_offset = ‘yes’ as well as invert.solve\_for\_InSAR\_ramp = ‘no’, then a ramp will be solved for instead of an offset.

invert.regularise\_moment = ‘no’

String. 'yes' or 'no'. This adds an M0 prior likelihood - a normal distribution, with mean and std given in data.seismic\_moment and data.moment\_std

invert.alpha2\_initial = 1

Number. One number for each fault strand, of initial alpha^2 (variance) value. This choice of number should not affect the final result, but choosing well may reduce the burn-in time. If the hyperparameter is drastically wrong this can cause the inversion to fail.

invert.alpha2\_step\_size = 0.01

Number. Initial step size of alpha2\_modelparameter, which is then adjusted through the inversion.

invert.probability\_target\_initial = 0.001

Number. Initial probability target - if we add half the step size to a parameter, this is the probability we're aiming the perturbation to make. This is adjusted through the iteration process: if the rejection rate is too high, then the probability decreases to try to decrease step sizes.

invert.solve\_for\_beta = ‘no’

String. 'yes' or 'no' - beta is a hyperparameter on the data. Default is 'no'.

invert.beta\_initial = 1

Number. Starting value of hyperparameter that acts on the var-covariance matrix. Default is 1.

invert.beta\_step\_size = 0

Number. Starting value of hyperparmaeter stepsize. Default is 0.

invert.simulated\_annealing\_start = ‘no’

String. 'yes' or 'no'. slipBERI will perform an initial simulated annealing inversion, to use as the starting parameter for the Bayesian inversion.

invert.solve\_for\_fault\_size = ‘no’

String. 'yes' or 'no'

invert.add\_correlation\_matrix\_stabiliser = 'no'

String. 'yes' or 'no' Add a small term (akin to a nugget) to the diagonal of the von Karman matrix sigma\_s. Unlikely to be necessary unless solving for the fault size.

invert.load\_old\_MCMC\_chain = ‘no’

String. Either a name of a saved file, or 'no'. If this is a name of a file, then the MCMC chain will continue from the past max-likelihood solution, with same step sizes, alpha^2, and probability target. Important: inversion will continue with same parameters as the previous chain.

### **Priors**

Contains details on the priors that you wish to incorporate into your inversion.

priors.slip\_prior = ‘boxcar’

String. 'boxcar' or 'gaussian' or 'logarithmic', for Bayesian inversions Default choice is boxcar, i.e. there is equal probability of slip between your permitted minimum and maximum slip values (below). Note that if you use logarithmic, you will probably need to increase your number of iterations, since there are more rejections, and also maybe decrease your step size.

priors.min\_slip = 0

Number. Minimum value of slip allowed, in metres.

priors.max\_slip = 10

Number. Maximum value of slip allowed, in metres.

priors.predominant\_faulting\_style: {‘ss’}

String in curly brackets. Either {'ss'} for strike-slip or {'ds'} for dip-slip. This is used in von Karman smoothing to calculate the along-strike and down-dip correlation lengths.

priors.min\_rake = 70

Number. Minimum value of permitted rake. Rake 0 = left-lateral strike-slip. Rake 180 = right-lateral strike-slip. Rake 90 = thrust. Rake -90 (or 270) = normal. Can be a matrix of one value per each fault\_strand\_for\_smoothing, or if just one value used then same value is used across all fault strands.

priors.max\_rake = 110

Number. Maximum value of permitted rake. Rake 0 = left-lateral strike-slip. Rake 180 = right-lateral strike-slip. Rake 90 = thrust. Rake -90 (or 270) = normal. Can be a matrix of one value per each fault\_strand\_for\_smoothing, or if just one value used then same value is used across all fault strands.

priors.alpha2\_prior = ‘logarithmic’

String. 'logarithmic' or ‘boxcar’, depending on if you want to use a logarithmic prior on alpha2 or not. alpha^2 is a hyperparameter on the prior, see Amey et al. 2018 for details.

priors.min\_alpha2 = 0.001

Number. Minimum permitted alpha^2 hyperparameter. Note that if using logarithmic prior this is the minimum alpha^2 value, not maximum 10^(alpha^2) value.

priors.max\_alpha2 = 2

Number. Maximum permitted alpha^2 hyperparameter. Note that if using logarithmic prior this is the maximum alpha^2 value, not maximum 10^(alpha^2) value.

priors.alpha2\_flag = []

String. Either 'bothsame' if you wish to use the same alpha2 hyperparameter on multiple fault strands or ‘[]’ if not.

priors.max\_offset = 1

Number. Maximum permitted offset in meters if invert.solve\_forInSAR\_offset = 'yes'

priors.min\_offset = -1

Number. Minimum permitted offset in meters if invert.solve\_forInSAR\_offset = 'yes'

priors.min\_beta = 1

Number. Minimum value of permitted beta, which is a hyperparameter on the data. The default is to not solve for beta, so this value is only used if invert.solve\_for\_beta = ‘yes’

priors.max\_beta = 1

Number. Maximum value of permitted beta, which is a hyperparameter on the data. The default is to not solve for beta, so this value is only used if invert.solve\_for\_beta = ‘yes’

priors.min\_circharm\_coeffs = 0

Number. Minimum permitted circharm coefficient. This is related to the circular harmonics terms if solving for fault size, so this is only used if invert.sovle\_for\_fault\_size = ‘yes’

priors.max\_circharm\_coeffs = maximum\_coefficient

Number. Maximum permitted circharm coefficient. This is related to the circular harmonics terms if solving for fault size, so this is only used if invert.sovle\_for\_fault\_size = ‘yes’

priors.min\_circharm\_phi = -2\*pi

Number. Minimum permitted circharm rotation (radians). This is related to the circular harmonics terms if solving for fault size, so this is only used if invert.sovle\_for\_fault\_size = ‘yes’

priors.max\_circharm\_phi = 2\*pi

Number. Maximum permitted circharm rotation (radians). This is related to the circular harmonics terms if solving for fault size, so this is only used if invert.sovle\_for\_fault\_size = ‘yes’

priors.min\_circharm\_center = 0

Number. Maximum permitted circharm center location (m). This is related to the circular harmonics terms if solving for fault size, so this is only used if invert.sovle\_for\_fault\_size = ‘yes’

priors.max\_circharm\_center = [length\_of\_fault, width\_of\_fault]

Number. Maximum permitted circharm center location (m). This is related to the circular harmonics terms if solving for fault size, so this is only used if invert.sovle\_for\_fault\_size = ‘yes’

### **Elastic\_params**

Structure with details of elastic parameters to use for the elastic halfspace.

elastic\_params.lambda = 3.23e10

Number. Value of Lamé’s first parameter.

elastic\_params.mu\_okada = 3.23e10

Number. Value of Lamé’s second parameter, the shear modulus.

### **Display**

Structure with details of which parameters to plot at the end of the inversion.

display.plot\_resolution\_matrix = ‘yes’

String, 'yes' or 'no', whether to plot the resolution matrix before commencing the inversion.

display.plotmean = ‘yes’

String, 'yes' or 'no' to plot the mean and standard deviation or not.

display.plotmode = ‘yes’

String, 'yes' or 'no' to plot the mode and standard deviation or not.

display.plotmedian = ‘yes’

String, 'yes' or 'no'to plot the median and standard deviation or not.

display.plotMAP = ‘yes’

String, 'yes' or 'no', whether to plot the MAP (max a posteriori) and standard deviation or not

display.plotallsips = ‘yes’

String, 'yes' or 'no' to plot with the mean, mode, median, max likelihood in one plot (and true value, if if in testing mode).

display.plotprob = ‘yes’

String, 'yes' or 'no', whether to plot how the probability changes throughout the inversion.

display.plothists = ‘yes’

String, 'plothistall' or ‘plothistsample', whether to plot histograms for all the slip patches, or randomly select some to display.

display.plotsurfacedisp = ‘yes’

String, 'yes' or 'no', whether to display surface displacement, model and residual.

display.plotmarginalPDFs = ‘yes’

String, 'yes' or 'no', whether to plot the marginal PDFs for the six patches with highest slip.

display.plot3d = ‘yes’

String,'yes' or 'no', whether to plot the fault geometry with slip mode.

display.calc\_confidence = ‘no’

String. 'yes' or 'no' for whether to plot the 95% confidence intervals for each patch.

### **Housekeeping**

Choice of savename, which will be at the start of the outputted .mat file

housekeeping.save\_name = ‘slip’

String. name you'd like to call your data run at the end of the inversion. Note that slipBERI automatically appends it with your inversion style. So your output file will be called ‘[housekeeping.savename]\_parameters\_used\_in\_inversion.mat’

**Set up the structures**

To set-up these structures, after you have edited the file with the correct details, run this command in the Matlab terminal:

>> make\_structures\_required\_for\_slipBERI.m

## **Data errors**

To quantify the spatially correlated error in InSAR scenes slipBERI calculates a variance-covariance matrix using an exponential function (see section 4.1 of Amey et al., 2018).

This function requires the sill, nugget and range from a semivariogram on an undeforming region of the data (and which should be found in that order in the file listed in data.varcovar\_details).

GBIS (Barnardi and Hooper, 2018) provides nice codes to calculate the variance-covariance, and the user can create a text file with the sill, nugget and range calculated.

For further reading I direct the reader to:

Lohman & Simons, 2005

Oliver & Webster, 2014

Foody & Atkinson, 2002

Curran, 1988

## **Running the inversion**

### **Start the inversion**

To start the inversion, in the matlab terminal run the slipBERI function as follows:

>> slipBERI(fault, data, invert, priors, elastic\_params, display, housekeeping)

Each of these structures is set up automatically by the script:

make\_structures\_required\_for\_slipBERI

For which you should input the correct details for your inversion, using the section above for guidance.

### **Initial plots and data check**

If you have selected invert.quickcheck = ‘yes’ (recommended), then once all the data has been ingested and the fault model set up then some plots will appear on the screen, along with this on the Matlab terminal:

Is all your data and fault in the right place?

If so, click anywhere on the figure to continue.

If not you'd best sort that out...

This is a chance for you to check the location of your InSAR, GPS and fault before the inversion starts.

If everything is in the correct place, click on the figure and the inversion will continue.

### **During the inversion**

Once the inversion has started this will appear in the Matlab terminal:

Starting MCMC - Von Karman regularised Bayesian inversion

Followed by many lines like this:

Current rejection rate = 0.81 with 19 posterior accepts after 62 true trials (100 iterations)

At intervals throughout the inversion the rejection rate is calculated. For the first 10,000 iterations the step size for each model parameter is adjusted to try to get an ideal rejection ratio of 0.77 (Roberts et al., 1997). These first 10,000 iterations are then removed as the burn-in. If you find that the rejection ratio is very high and therefore most of the trials are being rejected, then you can change the parameters:

invert.step\_size

invert.alpha2\_step\_size

invert.probability\_target\_initial

If the rejection rate is very high (0.99) the inversion should in theory still converge, through it will require more iterations. However it’s worth checking the step sizes to ensure that they’re not very small, which would mean parameter space was not being fully explored.

If the rejection rate is 1 then there will most likely be a problem with your set-up or priors (e.g. only permitting left-lateral strike slip motion on a right-lateral strike slip fault).

Setting the correct step sizes and probability target can be difficult. If you wish to fix the step sizes instead, then comment out lines 2741-2743:

% Calculate new step sizes

if singularflag == 0 & i < 10000

[new\_step\_sizes] = calculate\_step\_sizes( ...

end

And uncomment line 2744:

%new\_step\_sizes = step\_sizes;

The progress of the inversion can be tracked by these messages:

One quarter of the way through…

Halfway through…

Three quarters through…

The number of iterations is also printed, so you can see how close to finishing the inversion is.

## **At the end of the inversion**

Once the inversion is finished this will be printed to the screen:

Almost done. Tidy tidy tidy.

.....Okay, enough of that now.

Displaying result.

slipBERI completed in 1581.4086 seconds. You're welcome.

At this point the figures the appropriate plots from the display structure will be plotted and the final solution saved in a ‘withburnin\_yourchosenname.mat’ file.

This will be displayed to the screen:

IMPORTANT: no burn in has been removed.

Using the probability plot and the histograms, decide how much burn in to remove

Set number of iterations to remove to burn\_in\_remove\_number in the terminal

Then run script 'remove\_burn\_in' in the terminal

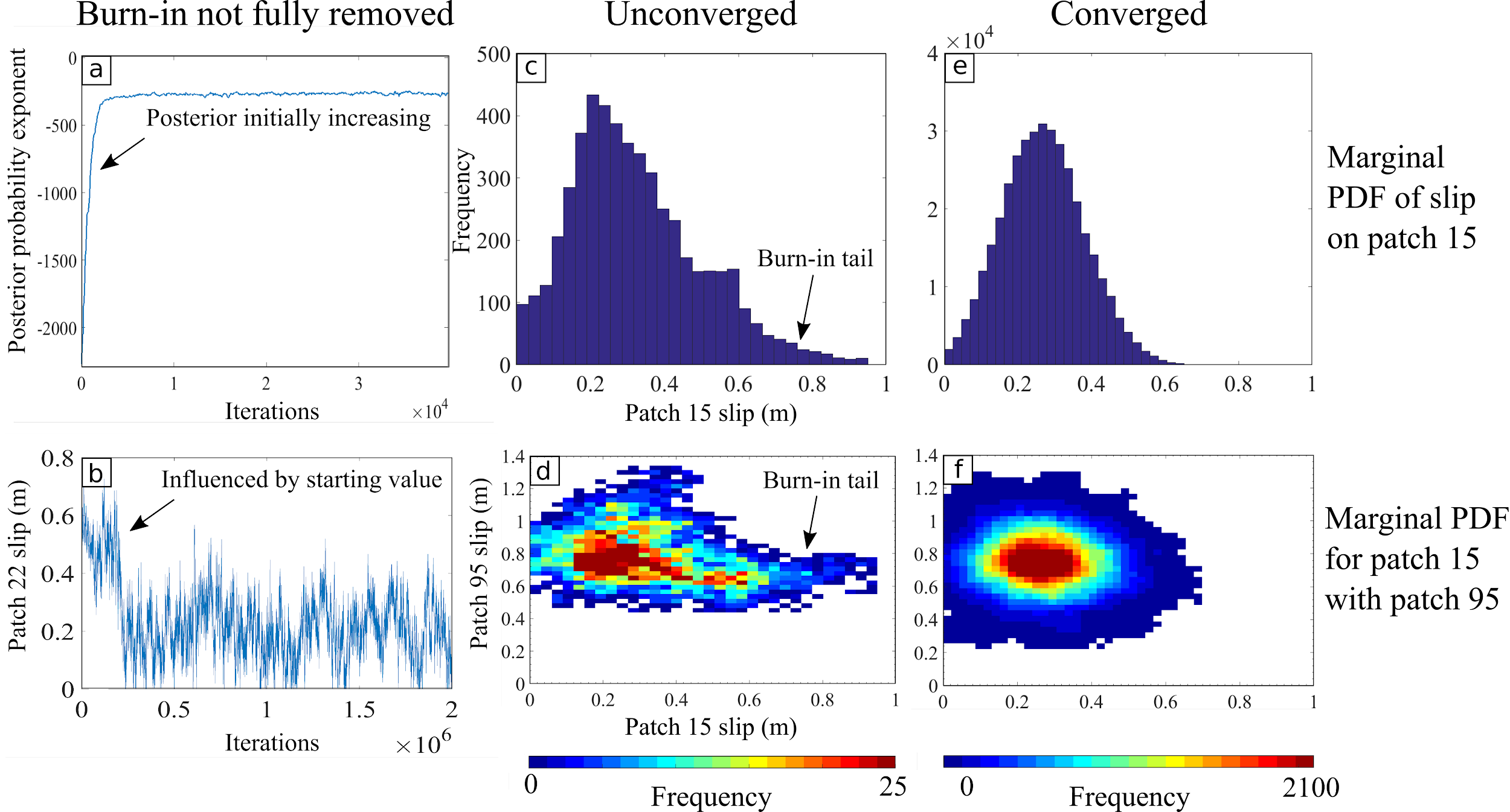
e.g.

burn\_in\_remove\_number = 10000

remove\_burn\_in

IMPORTANT: remove burn in now!

The initial saved solution has not removed any burn in. You must decide, based on the plots, how much burn in to remove. Criteria for choosing is shown in Figure 6 from Amey et al., 2018 (below).



In general, removing 10,000 - 20,000 iterations should be enough. Once you have decided run these commands in the terminal:

>> burn\_in\_remove\_number = 10000

>> remove\_burn\_in

This script will then remove the stated amount of burn in, resave the .mat file and rerun display\_result. You can then see if enough burn in has been removed.

The function will then pause at the end, giving chance to examine any of the matrices if desired.

This is displayed to the screen with instructions on how to exit the function:

Keyboard mode now. To terminate keyboard mode and end the slipBERI function, type 'dbcont' and press Enter

## **Errors**

\*\*\*

**Error:**

Current rejection rate = 0.99

Current rejection rate = 0.99

Current rejection rate = 0.99

Current rejection rate = 0.99

Not strictly an error, but if you are running an inversion and the rejection rate is very high (i.e. 99% of trials are being rejected), then it is best to stop the inversion. Technically the inversion is still working, and a solution will eventually converge. But the emphasis is on ‘eventually’ - it is not running efficiently, and will take an extremely long time.

Assuming that all of your data is correct (check utm/ latlong issues, your data and errors are the correct units, etc, etc) then it is a matter of fine-tuning the initial parameters.

In the ‘invert’ structure there are a number of parameters that can be fine-tuned:

slip\_initial:

step\_size:

alpha2\_initial:

alpha2\_step\_size:

probability\_target\_initial:

As well as others, if you are solving for beta, fault size, etc.

As a general rule, if my rejection rate is too high, make step sizes smaller. There is a balance, as if your step sizes are too small the search will not be complete. E.g. if your slip step size is ridiculously small, such as [±](https://howtotypeanything.com/plus-minus-symbol/) 1mm, you will need to increase the number of iterations to ensure a full search of parameter space has been done.

As a rule of thumb, slipBERI is most sensitive to alpha2 and probability\_target\_initial, and then to step\_size. I would not spend mental effort tinkering with slip\_initial, as long as your initial estimates are not wild (50m), then the correct slip should be found whatever your starting parameters.

There are methods to get around having to fine-tune these parameters, such as ensemble sampling - I direct the reader to [Foreman et al 2012](https://arxiv.org/pdf/1202.3665.pdf). This ensemble sampling is actually coded into slipBERI and should work for a one fault-strand set-up. However, whilst I found this work fantastically well for a small number of slip patches, it did not work for the number of slip patches that I have used in the publications. I invite you to solve this!

\*\*\*

**Error message:**

Error setting property 'Ticks' of class 'ColorBar':

Value must be a vector of type single or double whose values increase

Error in matlab.graphics.illustration.ColorBar/set.XTick

Error in plotmatrix\_lower (line 440)

set(cc,'XTick',[0 xlabel\_pos],'xticklabel',num2str([0 xlabel\_pos]'))

Error in display\_result (line 1460)

[H,AX,BigAx,P,PAx,cc\_colorbar] =

plotmatrix\_lower(slip\_for\_marginal\_PDF\_plotting,'plot\_color'); % with many thanks

to David Bekaert for this script % [H,AX,BigAx,P,PAx,cc\_colorbar] =

plotmatrix\_lower\_david(data(ix\_range\_3,:),'plot\_color',data\_opt);

Error in run (line 91)

evalin('caller', strcat(script, ';'));

Error in slipBERI (line 2996)

run('display\_result.m');

**Cause:** The inversion has been run with too few iterations, so plotting the 2D histograms in ‘display result’ has failed. The solution will still have saved and the other figures should plot okay.

**Solution:** Run inversion with more iterations. Or set

display.plotmarginalPDFs = ‘no’

and run display\_result again

\*\*\*

**Error:**

DANGER: A rogue NaN has been located in calc\_loglikely

**Possible causes:**

## **References**

Amey, R. M. J., Hooper, A., & Walters, R. J. (2018). A Bayesian method for incorporating self‐similarity into earthquake slip inversions. *Journal of Geophysical Research: Solid Earth*, 123, 6052–6071. <https://doi.org/10.1029/2017JB015316>

Amey, R. M. J., Hooper, A., & Morishita, Y. ( 2019). Going to any lengths: Solving for fault size and fractal slip for the 2016, *Mw* 6.2 Central Tottori earthquake, Japan, using a transdimensional inversion scheme. *Journal of Geophysical Research: Solid Earth*, 124.<https://doi.org/10.1029/2018JB016434>

Bagnardi M. & Hooper A, (2018). Inversion of surface deformation data for rapid estimates of source parameters and uncertainties: A Bayesian approach. Geochemistry, Geophysics, Geosystems, 19. <https://doi.org/10.1029/2018GC007585>

Available for download here: <https://comet.nerc.ac.uk/gbis/>

Candela, T., F. Renard, Y. Klinger, K. Mair, J. Schmittbuhl, and E. E. Brodsky (2012), Roughness of fault surfaces over nine decades of length scales, *J. Geophys. Res.*, 117, B08409, doi:[10.1029/2011JB009041](https://doi.org/10.1029/2011JB009041)

Curran, P. (1988). The semivariogram in remote sensing: An introduction. *Remote Sensing of Environment*, 24(3), 493–507.<https://doi.org/10.1016/0034-4257(88)90021-1>

Atkinson, P.M. and Foody, G.M., 2002. Uncertainty in remote sensing and GIS: fundamentals. *Uncertainty in remote sensing and GIS*, pp.1-18

Foreman-Mackey, D., Hogg, D.W., Lang, D. and Goodman, J., 2013. emcee: the MCMC hammer. *Publications of the Astronomical Society of the Pacific*, *125*(925), p.306.

Fukuda, J.I. and Johnson, K.M., 2008. A fully Bayesian inversion for spatial distribution of fault slip with objective smoothing. *Bulletin of the Seismological Society of America*, *98*(3), pp.1128-1146

Hastings, W.K., 1970. Monte Carlo sampling methods using Markov chains and their applications.

Hooper, A.J., 2012, December. Earthquake slip distribution estimation, using a random vector approach. In *AGU Fall Meeting Abstracts*

Lohman, R.B. and Simons, M., 2005. Some thoughts on the use of InSAR data to constrain models of surface deformation: Noise structure and data downsampling. *Geochemistry, Geophysics, Geosystems*, *6*(1)

Mai, P.M. and Beroza, G.C., 2002. A spatial random field model to characterize complexity in earthquake slip. *Journal of Geophysical Research: Solid Earth*, *107*(B11), pp.ESE-10.

Metropolis, N., Rosenbluth, A.W., Rosenbluth, M.N., Teller, A.H. and Teller, E., 1953. Equation of state calculations by fast computing machines. *The journal of chemical physics*, *21*(6), pp.1087-1092.

Milliner, C. W. D., J. F. Dolan, J. Hollingsworth, S. Leprince, F. Ayoub, and C. G. Sammis (2015), Quantifying near‐field and off‐fault deformation patterns of the 1992 Mw 7.3 Landers earthquake, *Geochem. Geophys. Geosyst.*, 16, 1577–1598, doi:[10.1002/2014GC005693](https://doi.org/10.1002/2014GC005693)

Mosegaard, K., and A. Tarantola (1995), Monte Carlo sampling of solutions to inverse problems, *J. Geophys. Res.*, 100(B7), 12431–12447, doi:[10.1029/94JB03097](https://doi.org/10.1029/94JB03097).

Oliver, M., & Webster, R. (2014). A tutorial guide to geostatistics: Computing and modelling variograms and kriging. *CATENA*, 113, 56–69.<https://doi.org/10.1016/J.CATENA.2013.09.006>

Roberts, G. O., Gelman, A., & Gilks, W. R. (1997). Weak convergence and optimal scaling of random walk metropolis algorithms. *The Annals of Applied Probability*, 7(1), 110–120.<https://doi.org/10.1214/aoap/1034625254>

Sagy A., Brodsky E. E., Axen G.J.; Evolution of fault-surface roughness with slip. *Geology* ; 35 (3): 283–286. doi:<https://doi.org/10.1130/G23235A.1>

Tarantola, A. (2005). Inverse problem theory and methods for model parameter estimation (2nd ed., pp. 50–52). Philadelphia: Society for Industrial and Applied Mathematics.