Electronic supplement to

Robust, fast, probabilistic source inversion using pattern recognition

Paul Käufl, Andrew P. Valentine, Ralph de Wit, Jeannot Trampert

July 14, 2015

The sections "Preprocessing & Input Vector Formation", "Neural network architecture and implementation details" and "Limitations, Computational Cost & Training Set Size" of this electronic supplement to the article provide additional information on the neural network methodology and implementation details. Supporting tables and figures regarding the observed dataset, source and earth models used throughout the article are provided in the section "Supporting Tables and Figure".

Preprocessing & Input Vector Formation

In the following $u_{j,k,l}$ denotes either a synthetic or a digitized, seismometer-corrected observed time-series, recorded at receiver j, for component $k \in 1, 2, 3$, at time-step l. Initially, we low-pass filter and downsample $u_{j,k,l}$ to the Nyquist sampling rate by integer decimation. The downsampling serves as a lossless compression which in addition avoids an artificial temporal correlation of the observational noise $\boldsymbol{\epsilon}_d$.

We denote by t_0 the time at which the first signal is detected at the station closest to the epicenter. From each trace in the (observed or synthetic) dataset, we select the time window given by $[t_0 + \delta_0, t_0 + \delta_0 + \Delta T]$. A small random time shift δ_0 is added to account for uncertainties in the determination of the onset-times. Note that while we are in theory flexible to choose more complex windowing schemes, e.g. multiple, disjoint windows in order to select different phases independently and allow for relative travel-time shifts, the length of the windows has to stay fixed over the dataset, in order for the network input vectors to be of constant length.

Let $u_{j,k,l}$ from now on denote the filtered, downsampled and windowed timeseries. In the case of synthetic traces, a noise component $\epsilon_{j,k,l}$, drawn from a noise distribution accounting for any unmodelled part of the observed signal is subsequently added, giving

$$u^{\dagger}_{j,k,l} = u_{j,k,l} + \epsilon_{j,k,l}. \tag{S1}$$

We transform the 3-component traces from cartesian (north-east-up) coordinates into a local spherical coordinate system as follows

$$u^{\dagger\dagger}_{j,1,l} = \ln\left(\sqrt{\sum_k \left(u^{\dagger}_{j,k,l}\right)^2}\right), \tag{S2a}$$

$$u^{\dagger\dagger}_{j,2,l} = \arccos\left(\frac{u^{\dagger}_{j,1,l}}{u^{\dagger\dagger}_{j,1,l}}\right), \tag{S2b}$$

$$u^{\dagger\dagger}_{j,3,l} = \operatorname{arctan2}\left(u^{\dagger}_{j,2,l}, u^{\dagger}_{j,3,l}\right). \tag{S2c}$$

Furthermore, the quantities

$$\overline{u}_{j,k} = \frac{\sum_l u^{\dagger\dagger}_{j,k,l}}{L}, \tag{S3a}$$

$$a_{j,k} = \max_l(|u^{\dagger\dagger}_{j,k,l} - \overline{u}_{j,k}|) \tag{S3b}$$

are calculated and the traces subsequently normalised using

$$\hat{u}_{j,k,l} = \frac{u^{\dagger\dagger}_{j,k,l} - \overline{u}_{j,k}}{a_{j,k}}. \tag{S4}$$

Finally, the timeseries are arranged into a joint data vector $\mathbf{d}$, by concatenating

$$\mathbf{d} = (\hat{\mathbf{u}}, \overline{\mathbf{u}}, \mathbf{a}). \tag{S5}$$

Note that by appending the means $\overline{\mathbf{u}}$ and scales $\mathbf{a}$ to the input vector, we provide all information necessary to reconstruct the original noisy traces $u^{\dagger\dagger}_{j,k,l}$ from the transformed input vectors and therefore no information is lost.

Neural network architecture and implementation details

Throughout this work we use ensembles of MDNs, which are described in detail in *Käufl et al.* (2014). All neural networks are trained using error back-propagation and weight updates are calculated using the L-BFGS quasi-newton method (*Nocedal*, 1980). From the set of prior samples 75% are used as a training set, 20% as validation set and 5% as a test set, which is not used during training. Each MDN is controlled by two parameters, the number of hidden units H, and the number of Gaussian kernels M. For each parameter and input-data type we train C MDNs, which are subsequently combined into an ensemble as described in *Käufl et al.* (2014). The number of hidden units is drawn randomly from a range for each MDN individually in order to further increase the variability of the members across the ensemble. The number of Gaussian kernels has been fixed to $M = 6$. Table S1 lists the choices for H and C for each input-data type and source parameter.

Limitations, Computational Cost & Training Set Size

While the present approach can in principle be used for a wide variety of input types and for relatively high-dimensional input spaces, the number of network parameters grows with the number of input components I. Thus restrictions are imposed on the dimensionality of data that can be used in this manner, within the limitations of available computational resources for neural network training. For example, in the case of 34 three-component recordings and a window length of 10 time-steps, a network with 10 hidden units would comprise $\mathcal{O}(10^4)$ network parameters. In addition, in order to achieve a constant compression ratio (that is number of hidden units per input node), the number of hidden units also has to be increased along with the input length. Therefore the number of network parameters will in the worst case grow as $\mathcal{O}(I^2)$. In order for the network parameters to be constrained during the training process it is necessary to provide a larger number of training patterns than parameters. Typically 10 to 20 times as many training examples as network weights are required to stabilize the

training procedure and achieve a good performance on the test set (*Bishop*, 1995). The L-BFGS method used in this work requires $\mathcal{O}(N \cdot M)$ operations per iteration, where N is the number of training examples and M the number of parameters. Therefore the training time increases quadratically with the number of network weights and thus — in the worst case — depends on the number of input nodes as $\mathcal{O}(I^4)$. This could potentially be lessened with the help of techniques such as mini-batch training, sparse network connectivity (*Bishop*, 1995) or less demanding training procedures. Another option is to reduce the input dimensionality beforehand by means of an additional feature extraction stage.

Note, however, that it is not essential to draw more samples from the prior distribution, if the number of input nodes is increased. The required amount of training examples can instead simply be generated by replicating a set of exisiting samples, but with different noise realization. That is, given a set of N samples $\{(\mathbf{m}_1, \mathbf{d}_1 + \boldsymbol{\epsilon}_1), \dots (\mathbf{m}_N, \mathbf{d}_N + \boldsymbol{\epsilon}_N)\}$, we generate a new set of $k \cdot N$ samples $\{(\mathbf{m}_1, \mathbf{d}_1 + \boldsymbol{\epsilon}_1), \dots (\mathbf{m}_1, \mathbf{d}_1 + \boldsymbol{\epsilon}_k), \dots (\mathbf{m}_2, \mathbf{d}_2 + \boldsymbol{\epsilon}_{2k}), \dots (\mathbf{m}_N, \mathbf{d}_N + \boldsymbol{\epsilon}_{2N})\}$. The effectiveness of such action can be understood, if we consider, that the fact that more information (e.g. longer time-windows, more stations, etc.) are incorporated in the inversion does not change the prior model space volume. The volume of the joint data-model space is increased, however, due to the additional input dimensions. By replicating the training set, i.e. increasing the number of data vector examples for any given source vector, we effectively increase the sampling density in the joint data-model space. For an example of networks trained with and without, respectively, training set replication, see Figure S1.

References

Bishop, C. M. (1995), *Neural Networks for Pattern Recognition*, vol. 92, 1642 pp., Oxford University Press.

Bock, Y., D. Melgar, and B. W. Crowell (2011), Real-time strong-motion broadband displacements from collocated GPS and accelerometers, *Bulletin of the Seismological Society of America*, *101*(6), 2904–2925.

Käufl, P., A. P. Valentine, T. B. O'Toole, and J. Trampert (2014), A framework for fast probabilistic centroid-moment-tensor determination–inversion of regional static displacement measurements, *Geophysical Journal International*, (3), 1676—-1693.

Melgar, D., Y. Bock, and B. W. Crowell (2012), Real-time centroid moment tensor determination for large earthquakes from local and regional displacement records, *Geophysical Journal International*, *188*(2), 703–718.

Nocedal, J. (1980), Updating Quasi-Newton Matrices with Limited Storage, *Mathematics of Computation*, *35*(151), 773.

Zheng, Y., J. Li, Z. Xie, and M. H. Ritzwoller (2012), 5Hz GPS seismology of the El Mayor-Cucapah earthquake: estimating the earthquake focal mechanism, *Geophysical Journal International*, *190*(3), 1723–1732.

Tables

Table S1: Neural network parameters.

	waveforms $\Delta T \in \{12\text{ s}, 24\text{ s}, 48\text{ s}\}$	$\Delta T = 60$ s	static displacements
κ	$H \in [15;30]$, $C = 5$	$H \in [25;40]$, $C = 15$	$H \in [20;50]$, $C = 30$
σ	$H \in [15;30]$, $C = 5$	$H \in [25;40]$, $C = 30$	$H \in [20;50]$, $C = 30$
h	$H \in [15;30]$, $C = 5$	$H \in [25;40]$, $C = 15$	$H \in [20;50]$, $C = 30$
M_w	$H \in [15;30]$, $C = 5$	$H \in [25;40]$, $C = 11$	$H \in [20;50]$, $C = 30$
depth	$H \in [15;30]$, $C = 5$	$H \in [25;40]$, $C = 15$	$H \in [20;50]$, $C = 30$
lat	$H \in [15;30]$, $C = 5$	$H \in [15;30]$, $C = 15$	$H \in [20;50]$, $C = 30$
lon	$H \in [15;30]$, $C = 5$	$H \in [15;30]$, $C = 15$	$H \in [20;50]$, $C = 30$
τ	$H \in [15;30]$, $C = 5$	$H \in [15;30]$, $C = 9$	$H \in [20;50]$, $C = 30$

Table S2: Reference double-couple solution for the 2010 Mw 7.2 El Mayor Cucapah event.

strike	rake	dip	M_w	depth	lat	lon	τ
52°	−14°	77°	7.2	10.0 km	32.23°	−115.39°	9.36 s

Table S3: 1-D layered crustal and upper mantle model for Southern California.

Layer	Thickness [km]	v_p [km/s]	v_s [km/s]	ρ [kg/cm^3]
1	1.0	2.0	1.0	2.0
2	0.5	4.0	2.0	2.0
3	9.0	6.0	4.0	3.0
4	8.0	6.0	4.0	3.0
5	8.0	7.0	4.0	3.0
6	10000	8.0	5.0	3.0

Figures

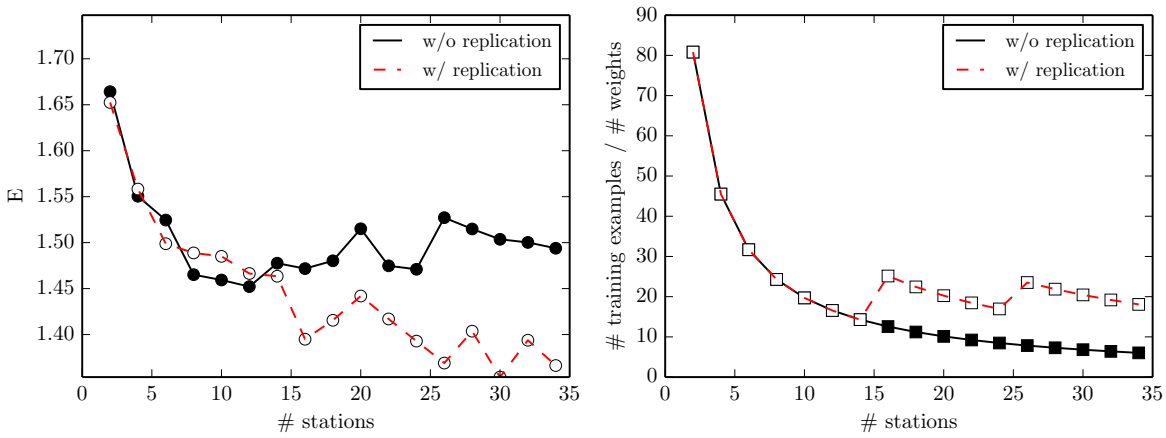

Figure S1: **Positive effect of training set replication. Left:** The prediction error E is defined as in *Käufl et al.* (2014) (equation 12) and is a measure for the precision and accuracy of the test set predictions. The test set error is plotted as a function of the number of stations in two cases. First, the number of training samples is kept constant (black, solid line). Second, it is scaled to the number of network parameters using replication (red, dashed line). The more data (stations) we use, the lower we expect the error to become, since potentially more information on the source parameters are available (due to increased azimuthal coverage, etc.). However, if we keep the number of samples in the training set constant (w/o replication), we in fact observe a slight increase in error. This effect is due to the lack of constraint on the increased number of neural network parameters (weights) due to the increase in input dimensions. If we, on the other hand, scale the number of training examples to the number of weights, to keep it (roughly) at 20 examples per weight, the effect is removed and the error drops with the number of stations as expected. **Right**: Number of examples in the training set divided by the number of network weights in the case that the number of training examples is kept constant (black, solid line) and scaled to the number of network parameters (red, dashed line). Note that we did not generate more source examples by solving the forward problem, but instead replicate the existing samples several times, each time drawing a different noise vector from the noise distribution. Networks were trained on κ using displacement waveform data and the number of hidden units has been fixed to 10.

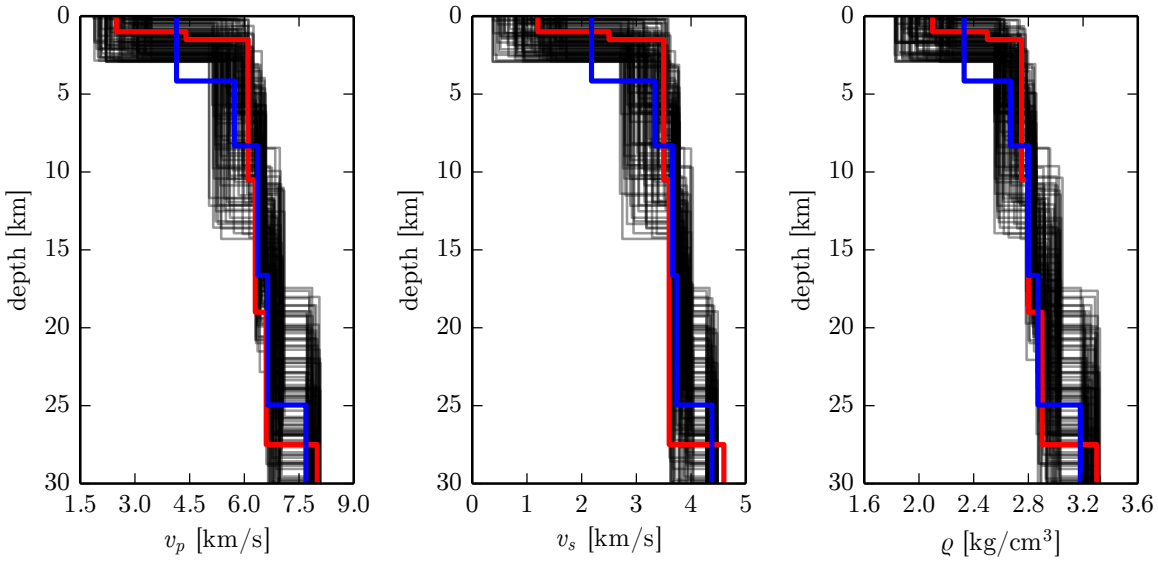

Figure S2: Test models drawn from the 1-D crustal model prior (see Table 2 in the manuscript). The thick, coloured lines show two models that have been previously used for local moment tensor inversions in Southern California by *Zheng et al.* (2012) (red) and *Melgar et al.* (2012) (blue), respectively, for reference. Note that the models extend uniformly into the upper mantle.

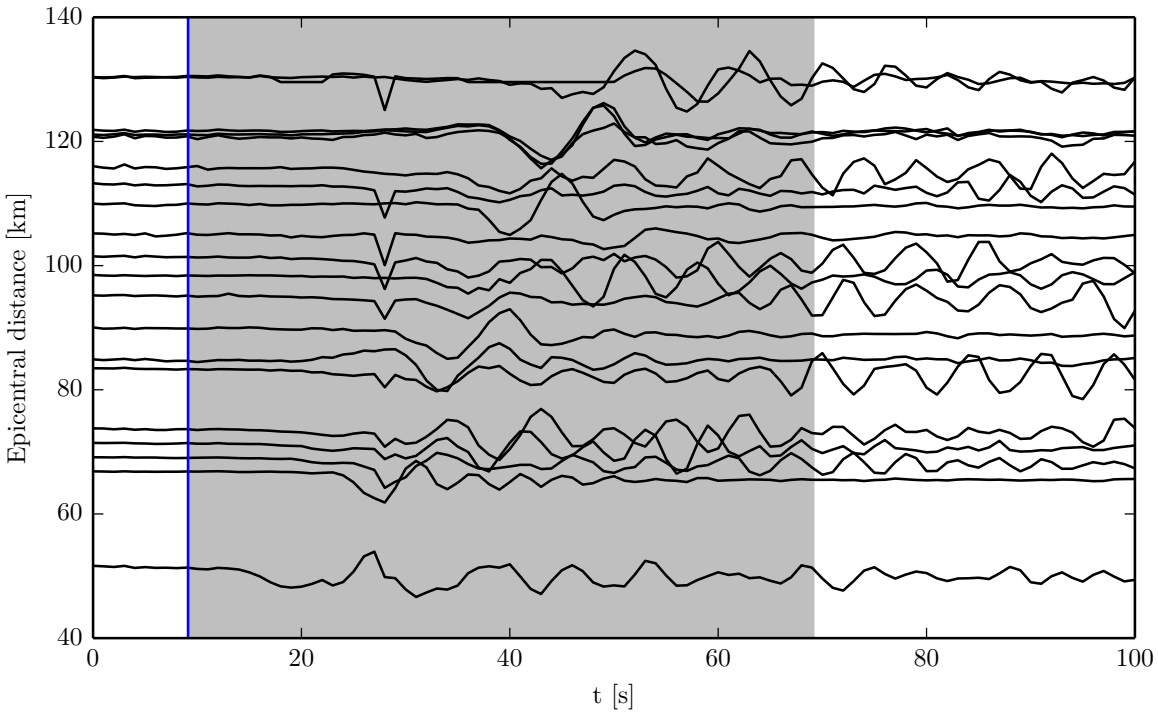

Figure S3: Normalized displacement waveforms recorded at 20 of the CRTN GPS receivers. The vertical blue line denotes the time t_0 and the lighter shaded area corresponds to a 60 s data window. The feature present in some of the waveforms shortly before 30 s is an artifact caused by variations in the GPS reference station during instantaneous positioning (*Bock et al.*, 2011).

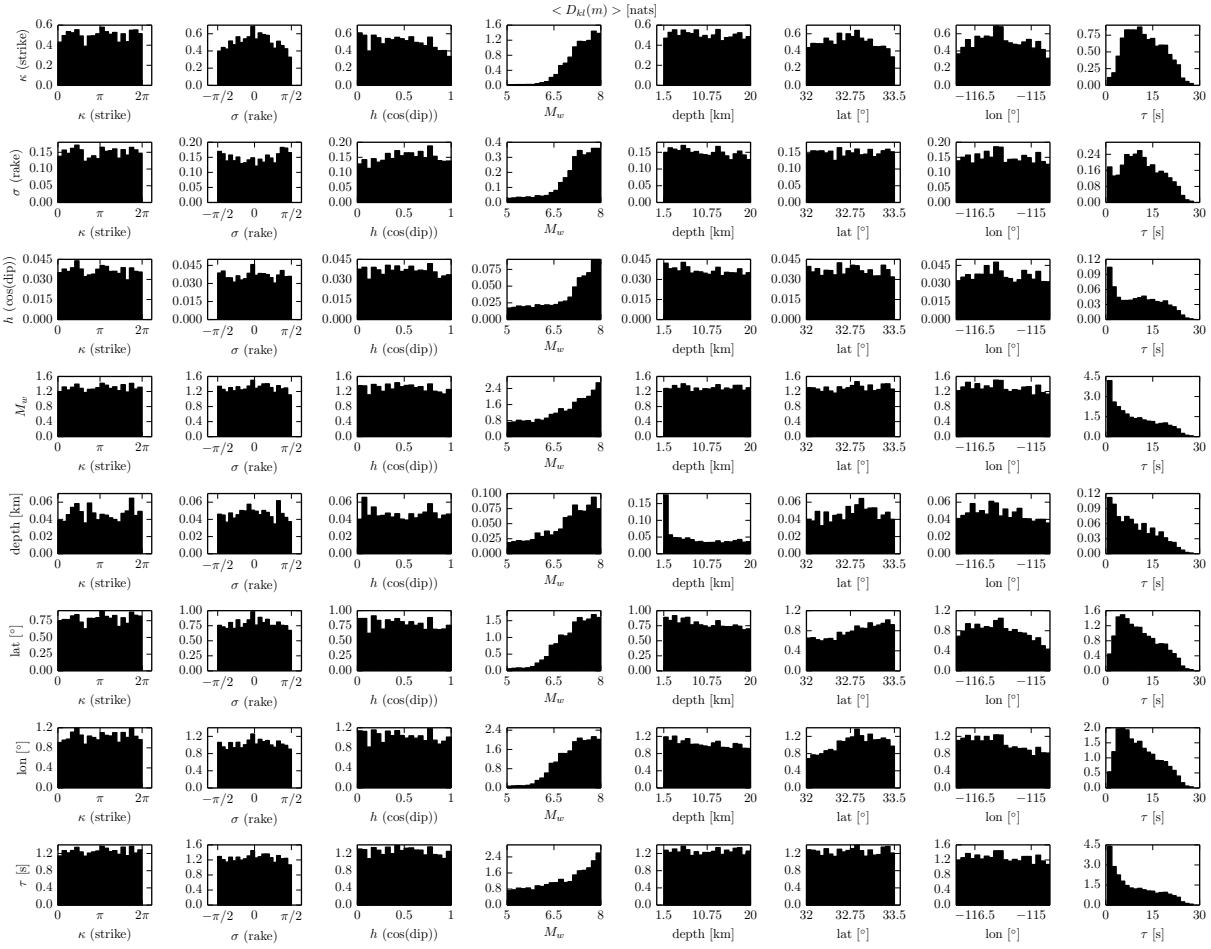

Figure S4: Histograms of parameter A (rows) weighted by the average information gain of all test set examples falling into the respective bin of the parameter B (columns). Units of the vertical axis are *nats*.