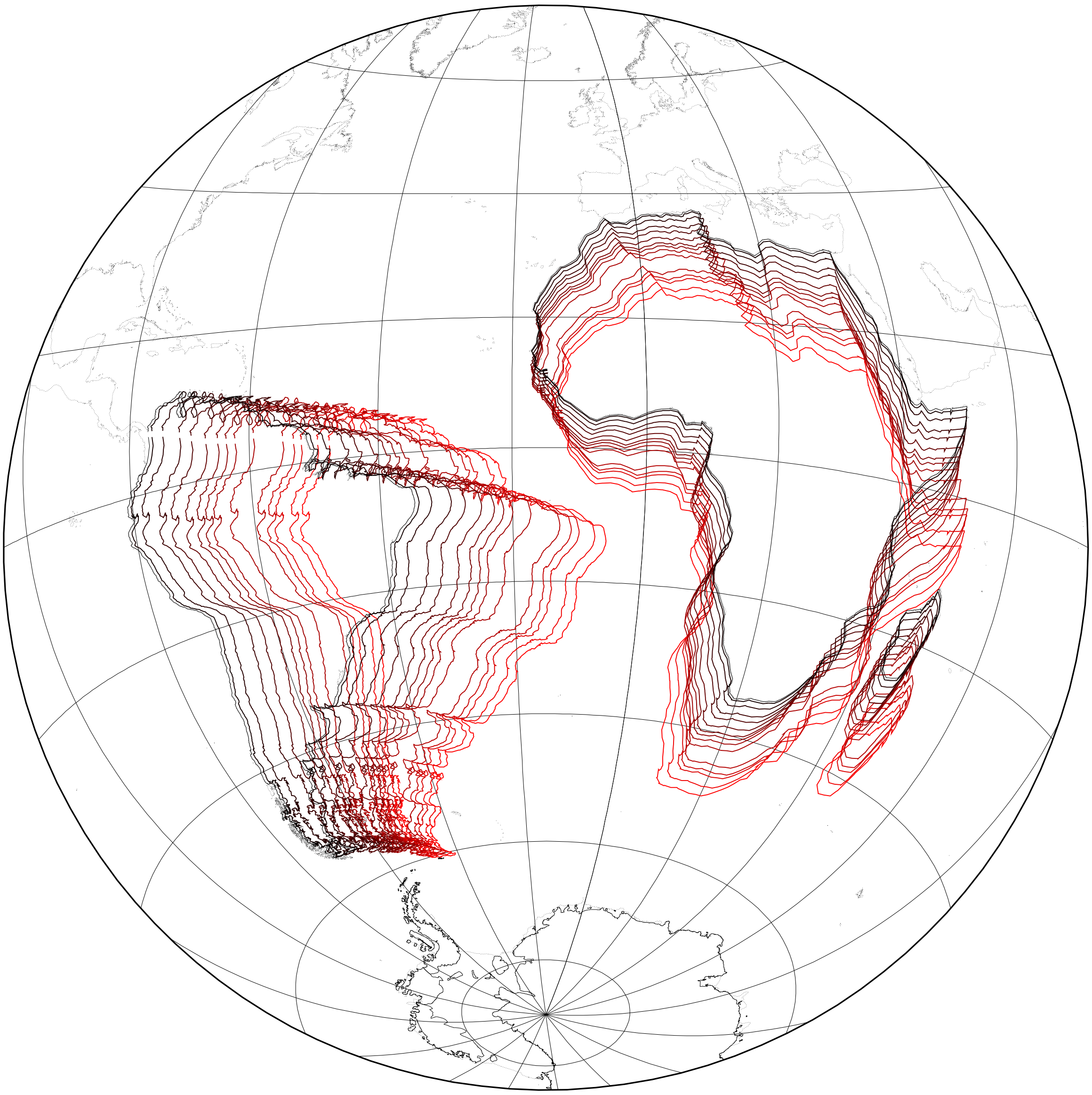
Inversion Suite
for 2 and 3-Plate Reconstructions

by

Adrian P. Nankivell



Contents

1

Inversion Theory

1.1 Method for a 2-plate reconstruction

Iterative inversion methods are based on determining how well a model fits a dataset, then making a change to this model which has been calculated to improve the fit of the dataset. The process is then repeated to find out how well this improved model fits the dataset, and so on, until a satisfactory fit has been obtained.

The first step is to define misfits that will give a quantitative measure of how well the model fits the dataset. In this case, different types of data are used, so a different misfit criterion is defined for each, which generates an error value (ε) for each datum, defined as displacements as opposed to distances, *i.e.* the error can be positive or negative. For example, the misfit for a fracture zone pick is defined as the perpendicular displacement of the pick from a synthetic flowline generated from the model parameters. If the pick lies above the flowline (Figure 1.5), then the error is positive, and negative if the pick is below the flowline. For magnetic anomaly picks, each pick is rotated towards a target great circle representing a different anomaly isochron (Figure 1.3), by a rotation determined from the model parameters which should in theory rotate the pick such that it lies exactly on the target great circle. The misfit is defined as the perpendicular displacement of this pick from the great circle; if the pick has been rotated beyond the great circle, the error is positive, and *vice versa.*

In order to determine how the model can be improved, the partial derivatives of these errors with respect to the model are calculated. In other words, for each datum, the effect of altering the model parameters is calculated, for each individual model parameter in turn. A value is calculated which represents how the error for the datum in question would change if that particular pole parameter was altered $\left(\frac{\partial\varepsilon}{\partial p^{\gamma}}\right)$, *e.g.* if the rotation angle is increased for one of the rotation poles, does this result in an increase or decrease in the error? By determining a complete set of errors, and a corresponding set of partial derivatives describing how each error will increase or decrease with a change to each of the model parameters, a least-squares inversion procedure will be able to determine a combination of changes to the model parameters which will reduce the overall magnitude of the errors of the dataset from the model.

The aim of the inversion is to determine a set of l finite rotation poles, comprising of a latitude, longitude and rotation angle (θ, ϕ, α) for each pole, giving a total of $n = 3l$ pole parameters, the set $\mathbf{p} = \{p_1, \dots, p_j, \dots, p_{3l}\}$. As input to the inversion, there are three sets of data: m^{fz} fracture zone location data points; m^{tf} transform fault azimuth data points; and m^{mag} magnetic isochron data points. Hypothetically there will be a perfect set of pole parameters $\mathbf{p}^*$, which will fit this combined dataset exactly. Then from the total set of $m(= m^{\mathrm{fz}} + m^{\mathrm{tf}} + m^{\mathrm{mag}})$ data points, consider the i^{th} data point, with error $\varepsilon_i(\mathbf{p})$ for the set of pole parameters $\mathbf{p}$, where for the perfect set of pole parameters

$$\varepsilon_i(\mathbf{p}^*) = 0 \tag{1.1}$$

For each data point, there exists a partial derivative relating the change in the error ε_i for a perturbation to the j^{th} pole parameter, which is given as $\frac{\partial \varepsilon_i}{\partial p_j}$. If we assume that the set of poles parameters $\mathbf{p}$ is close to the perfect set, $\mathbf{p}^*$, then

$$\varepsilon_i(\mathbf{p}^*) - \varepsilon_i(\mathbf{p}) \quad \approx \quad \frac{\partial \varepsilon_i}{\partial p_j}(p_j^* - p_j) \tag{1.2}$$

then defining the difference between the pole parameters as

$$\Delta p_j = p_j^* - p_j \tag{1.3}$$

and applying equation (1.1) we obtain

$$\varepsilon_i(\mathbf{p}) \approx -\sum_j \Delta p_j \frac{\partial \varepsilon_i}{\partial p_j} \tag{1.4}$$

To combine the three datasets on an equal basis, then we need to normalise the data, which is done by dividing each datum by its corresponding uncertainty. The uncertainties of the magnetics, fracture zone and transform data are defined as σ^{mag}, σ^{fz} and σ^{tf} respectively.

A normalised matrix of partial derivatives is then constructed for each data type,

$$\hat{\mathbf{A}}_{ij}^{\mathrm{mag}} = \frac{1}{\sigma^{\mathrm{mag}}} \frac{\partial \varepsilon_i^{\mathrm{mag}}}{\partial p_j} \qquad \hat{\mathbf{A}}_{ij}^{\mathrm{fz}} = \frac{1}{\sigma^{\mathrm{fz}}} \frac{\partial \varepsilon_i^{\mathrm{fz}}}{\partial p_j} \qquad \hat{\mathbf{A}}_{ij}^{\mathrm{tf}} = \frac{1}{\sigma^{\mathrm{tf}}} \frac{\partial \varepsilon_i^{\mathrm{tf}}}{\partial p_j} \tag{1.5}$$

Similarly vectors containing the normalised errors are constructed,

$$\hat{\mathbf{b}}_i^{\mathrm{mag}} = -\frac{1}{\sigma^{\mathrm{mag}}} \varepsilon_i^{\mathrm{mag}} \qquad \hat{\mathbf{b}}_i^{\mathrm{fz}} = -\frac{1}{\sigma^{\mathrm{fz}}} \varepsilon_i^{\mathrm{fz}} \qquad \hat{\mathbf{b}}_i^{\mathrm{tf}} = -\frac{1}{\sigma^{\mathrm{tf}}} \varepsilon_i^{\mathrm{tf}} \tag{1.6}$$

Finally, the matrices/vectors for all three datasets are then combined into a single normalised matrix $\hat{\mathbf{A}}$ of dimensions m by n and a single vector $\hat{\mathbf{b}}$ of length m,

$$\hat{\mathbf{A}} = \begin{bmatrix} \hat{\mathbf{A}}^{\mathrm{mag}} \\ \hat{\mathbf{A}}^{\mathrm{fz}} \\ \hat{\mathbf{A}}^{\mathrm{tf}} \end{bmatrix} \qquad \hat{\mathbf{b}} = \begin{bmatrix} \hat{\mathbf{b}}^{\mathrm{mag}} \\ \hat{\mathbf{b}}^{\mathrm{fz}} \\ \hat{\mathbf{b}}^{\mathrm{tf}} \end{bmatrix} \tag{1.7}$$

Then from equation (1.4) we have the matrix equation

$$\hat{\mathbf{A}}\,\Delta\mathbf{p} = \hat{\mathbf{b}} \tag{1.8}$$

However, in the inversion process, additional weighting is considered necessary in order to ensure that poor quality data have a lesser influence on the inversion and so, before solving this equation, a weighting matrix is applied. Whilst processing, various weighting factors are considered, detailed in Section 1.2 and a weighting matrix $\mathbf{W}$ of dimension m by m is constructed, given by

$$\mathbf{W}_{ij} = w_i \delta_{ij} \tag{1.9}$$

where δ_{ij} is the Kronecker delta tensor, and w_i is the scalar weight for the i'th datum (summation convention not applying).

So, prior to the inversion, $\mathbf{A}$ and $\mathbf{b}$ are premultiplied by the matrix $\mathbf{W}$, such that

$$\mathbf{A} = \mathbf{W}\hat{\mathbf{A}} \qquad \mathbf{b} = \mathbf{W}\hat{\mathbf{b}} \tag{1.10}$$

To avoid using up large amounts of memory and time constructing, and premultiplying by, such a large (diagonal) matrix, the input matrices are determined more simply as

$$\mathbf{A}_{ij} = w_i \hat{\mathbf{A}}_{ij} \qquad \mathbf{b}_i = w_i \hat{\mathbf{b}}_i \tag{1.11}$$

(summation convention not being relevant), and the inversion equation becomes

$$\mathbf{A}\,\Delta\mathbf{p} = \mathbf{b} \tag{1.12}$$

This is solved using a damped least squares method and from each iteration, a set of adjustments $\Delta\mathbf{p}$ to the current set of poles is produced, and after a number of inversions, a stable solution is found.

The complete theory and working for determining all of the errors and partial derivatives for all data types can be found in Adrian Nankivell's D.Phil Thesis (University of Oxford, 1997), but this is described in brief in the following sections.

1.1.1 Magnetic anomaly isochrons

The raw magnetic anomaly data come in the form of picks, consisting of a location in terms of longitude and latitude, and an anomaly identifier, such as c13. The data are then split into groups,

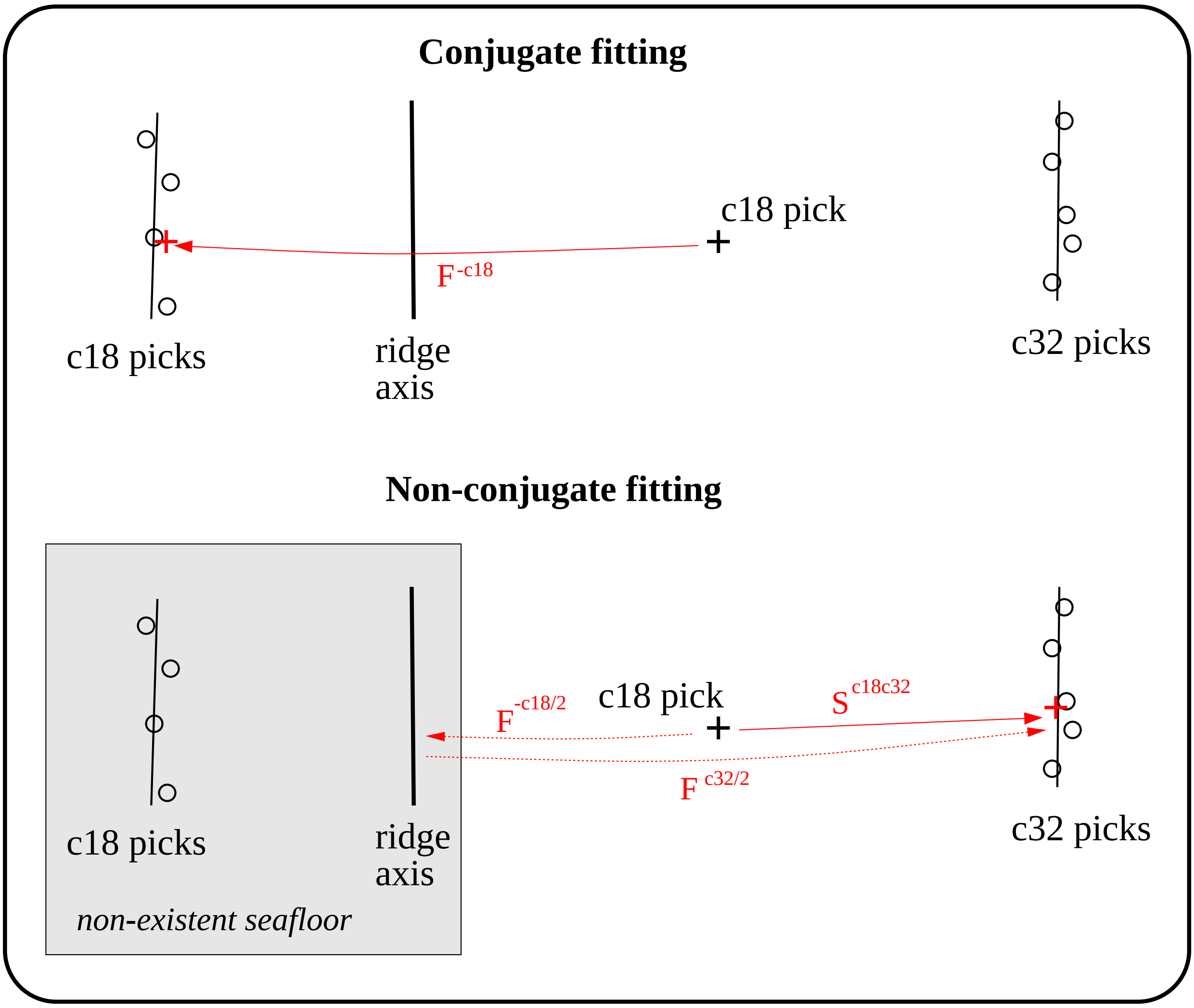


Figure 1.1: Diagram showing how in cases where seafloor has been subducted, it is not possible to rotate magnetic picks to their conjugate isochron, and must instead be fitted to picks of a different, non-conjugate, chron. **F** and **S** represent forms of finite and stage rotations respectively.

---

corresponding to anomalies created by the same continuous sections of mid-ocean ridge - *i.e.* not separated by a (significant) fracture zone or ridge offset.

This dataset is then preprocessed to provide input for the inversion, each ridge segment being considered separately. Initially, each anomaly group within the ridge segment with three or more picks has a great circle fitted to it using a singular value decomposition method. In some cases, where data is sparse, great circles are fitted to two picks only, from the vector product of the two vector representations of the pick locations (which gives a pole for the exact, unique great circle joining the two points). Then, for each set of anomaly picks, a target great circle (representing an isochron of another anomaly chron) is chosen for the data to be fitted to.

Ideally this will be the conjugate set of anomalies, but this is not possible in cases where one half of a plate pair is being, or has been, subducted and hence magnetic anomalies for many ages are only available on one side. Thus, picks must instead be fitted to great circles representing picks of a different chron, as shown in Figure 1.1, which adds some complexity to the calculations. In

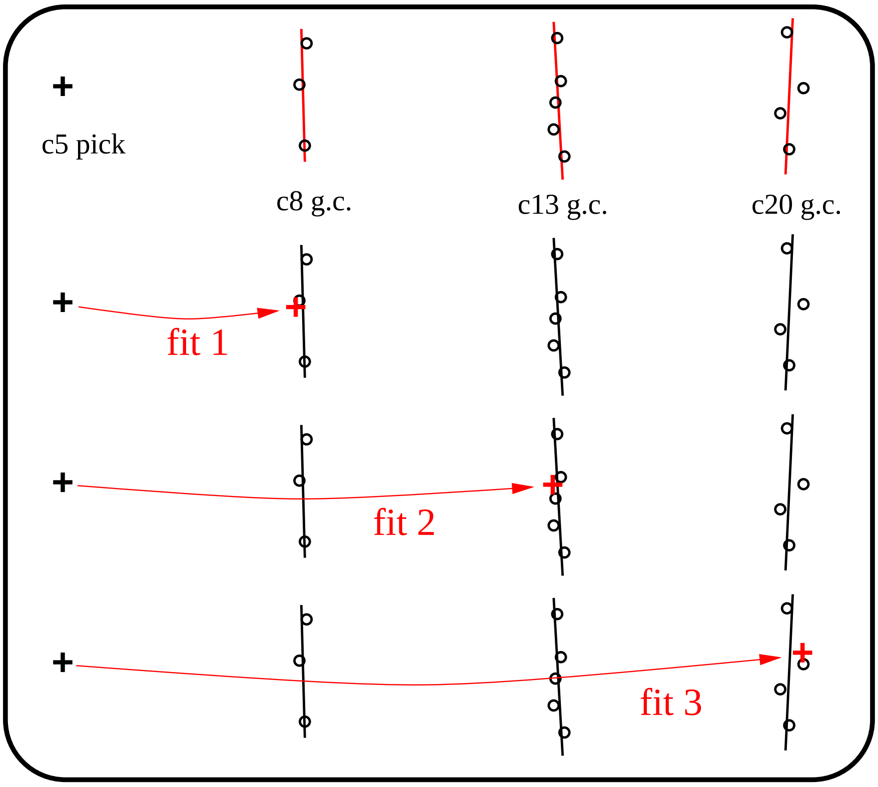


Figure 1.2: A diagram showing how when fitting anomaly picks to non-conjugate isochrons, several datum are generated from a single pick.

the case of conjugate fitting, the test rotation of the pick to the target great circle is derived from the finite rotation pole for that chron (either the actual finite rotation, or with a reversed rotation, depending on which side we are rotating the pick to). However, when rotating to a target great circle of a different chron, the test (stage) rotation is derived from the finite rotation poles for both chrons; it is a combination of a half rotation of the pick to the ridge axis, and then a half rotation from the ridge axis to the target great circle.

If an anomaly pick is not to be rotated to the conjugate anomaly, then a choice needs to be made as to which set of anomaly picks is to be used as the target great circle. The most sensible method is to rotate picks (without conjugate great circles) in turn to every great circle set within the same ridge section, as shown in Figure 1.2.

This generates several datum for each pick, *i.e.* if there are 3 great circles fitted in the ridge section, then an error and a set of partial derivatives are generated for each rotation and fitting in turn, leading to three sets of input data for each pick. A weight is applied to each non-conjugate generated datum, equal to the inverse of the number of great circles that the isochron pick is being rotated to. This ensures that overall contribution of a pick involved in non-conjugate fitting to the inversion is no greater than a pick involved in conjugate fitting, despite contributing a greater number of individual datum to the inversion. A feature of this method is that it allows picks for the central anomaly to be incorporated into the dataset, where the present day ridge exists. The disadvantage is that if asymmetric spreading or ridge jumps have occurred, then this will introduce

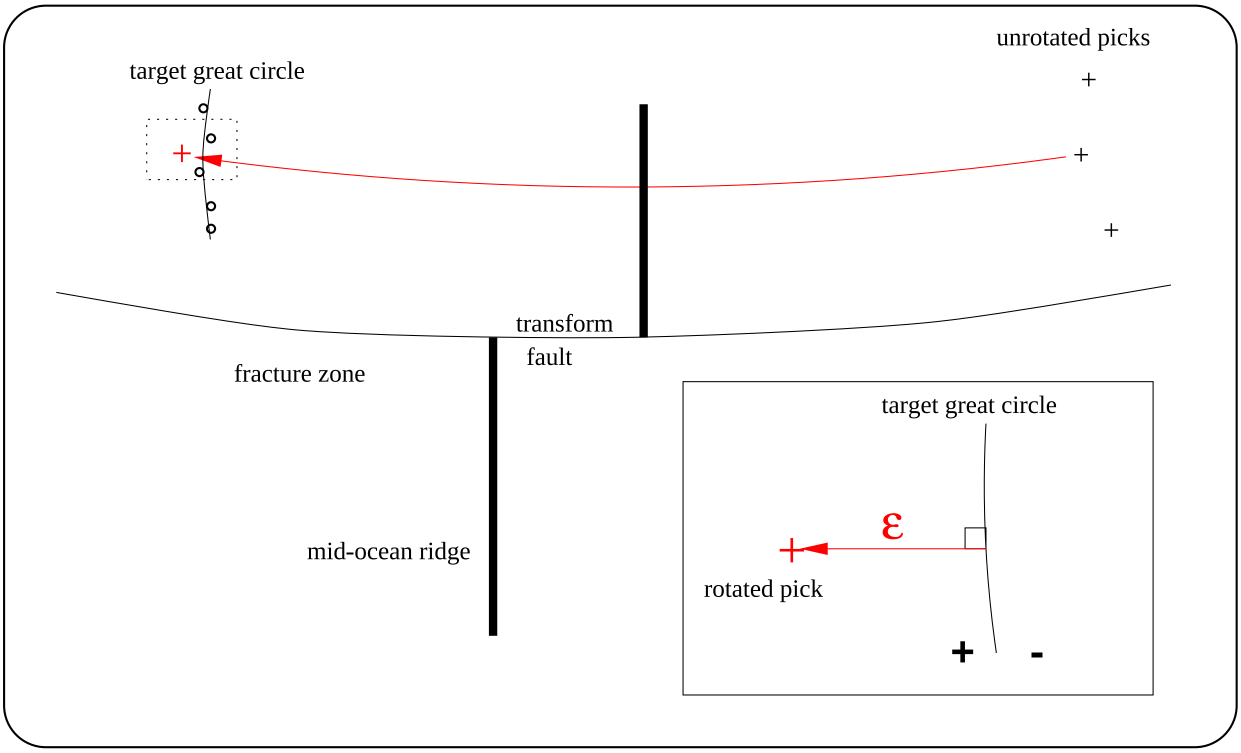


Figure 1.3: A diagram showing the misfit criteria for a magnetic pick when rotated to a target great circle

inherent errors into the problem, which does not occur when fitting to conjugate isochrons.

In the inversion procedure, each individual pick is rotated by the relevant 'stage' rotation towards the target great circle. The misfit is defined as the angular displacement of the rotated point from the closest point on the great circle, as shown in Figure 1.3. In fact, the notion of the closest point is only for figurative use, as the actual calculation of the error ε is carried out by finding the angular separation of the rotated point and the great circle pole, and subtracting 90 degrees.

For the inversion, the partial derivatives of the error are only calculated with respect to the pole parameters corresponding to the chrons of the current pick and the target great circle, naturally, altering any other pole parameter has no effect. In this way, the working is much simpler for the case where anomaly picks are matched to their conjugate set because then the stage rotation is solely defined by the 3 finite rotation parameters for that anomaly. When anomaly picks are being matched to the great circle of an anomaly group of differing age, then the stage rotation is a combination of two rotations, the first rotating the picks back to the present day ridge, or pseudo ridge, then another rotating the picks to the great circle, and so the the final stage rotation is dependent on a total of six finite rotation parameters.

1.1.2 Fracture zone locations

Each fracture zone is represented by a series of latitude, longitude picks, along with a single reference seed point. The seed point is used as a basis for calculating a flowline to which the

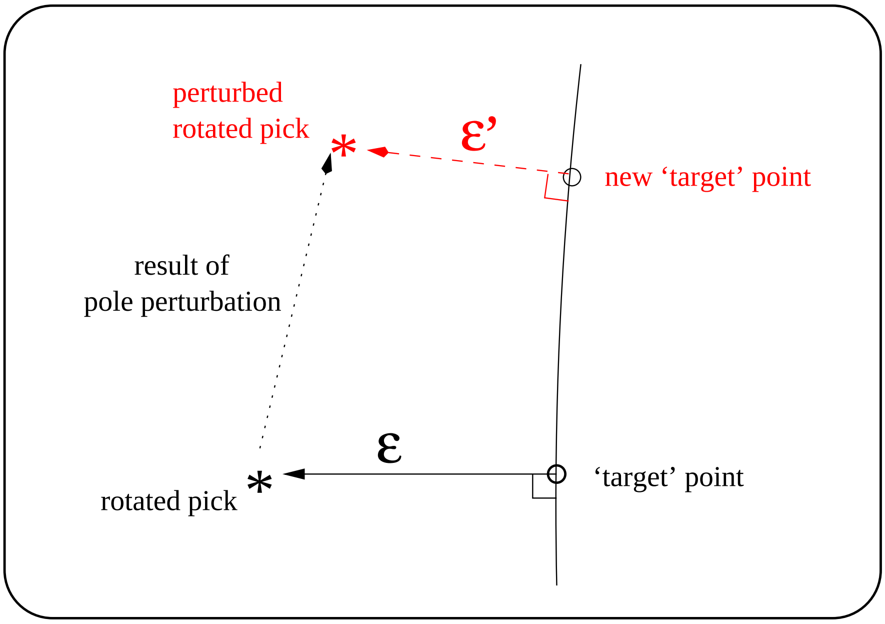


Figure 1.4: Diagram showing the effect of peturbing pole parameters on the errors when fitting magnetics picks to great circles.

picks are fitted. The ideal seed point is the fracture zone-ridge intersection, which represents the zero-age point along the flowline, but other reference points can be used, at any point along the fracture zone, as long as the age at the reference point corresponds to one of the chrons for which the inversion is being carried out. If the reference point is off-axis, the theoretical zero-age seed is calculated using the relevant half-rotation for the seed chron, and the flowline calculated from there. This approach is essential in cases where the mid-ocean ridge from which the fractures zones were generated is no longer in existence, and it is impossible to estimate the position of the ridge-transform intersections.

As an additional stage, the errors in the data for the fracture zone with respect to a perturbation in the reference point are calculated. The reference point is then shifted in a direction perpendicular to the flowline for the next stage in order to minimise the errors, i.e. if the reference point is the zero-age seed, then this is perturbed in a direction perpendicular to the current direction of plate motion. This compensates for any error in the estimation of the seed point, but this perturbation is limited to ±10km to avoid excessive seed shifts that might be compensating for a poor fit of the fracture zone to the synthetic flowline.

A synthetic flowline, based on the given seed point is calculated for the current rotation poles. The misfit for a fracture zone pick is given as the great circle displacement of the pick from the closest point along the flowline, as shown in Figure 1.5. As shown, the misfit for any given point is only affected by the location of the two flanking flowpoints, and hence on the two sets of 3 finite rotation parameters which were used to generate these flowpoints. The exception is for a pick lying within the section of the flowline between the zero-age point and the flowpoint defined by the first chron/pole, where the misfit is only affected by perturbations to the 3 parameters of the first finite rotation pole.

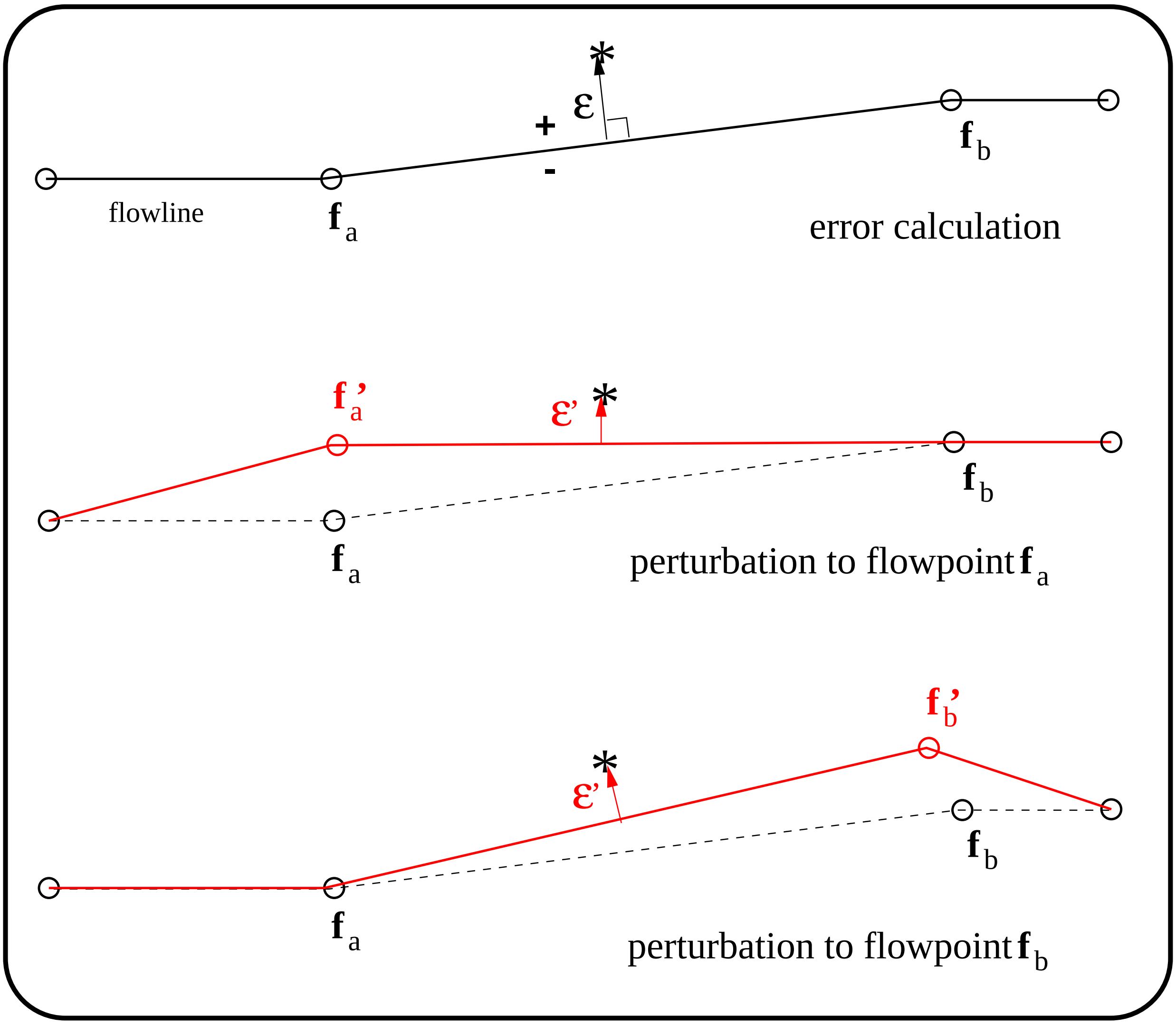


Figure 1.5: A diagram showing the misfit criteria for a fracture zone pick when fitted to a synthetic flowline, and the effect of perturbing the neighbouring flowpoints on the error.

1.1.3 Transform fault locations

In order to provide additional constraints on the first (effectively the current) rotation pole, it was decided to include transform fault data in the inversion. The strike of a transform fault is in theory a small circle defined by the current spreading pole. To determine the errors and partial derivatives for the transform picks, a small circle needs to be found for the transform in question, to which the data will be fitted. It could be generated by a single seed point, as for the fracture zone data, but instead, it is determined by finding the best fit small circle (of the current pole) for the transform picks.

The misfit for each pick along the transform is defined as simply the great circle distance between the pick and this best-fit small circle. This error can only be affected by a change in the latitude or the longitude of the current pole, it is unaffected by a change in the rotation angle of the current pole, or parameters of any of the other, older, poles.

1.2 Data weighting and selection

In addition to the normalisation of the data, some selection and basic weighting can be applied to the different data sets in order to guarantee that the inversion is not unduly influenced by data outliers or a particular set of inconsistent data. An additional factor is allowed, in that each data type is allocated a biasing factor, (β^{mag}, β^{fz}, β^{tf}), so that if desired, one can be favoured over the others.

The weighting schemes are now described, but it is important to note that the weighting factors are initially given as inverse weights, and converted into true weights at a later stage.

1.2.1 Magnetics data

For each magnetic anomaly group, the weighting is determined by four factors:

(i) the number of great circles, n_{gcs}, to which the current anomaly group is being rotated;

(ii) the RMS misfit ξ, of all the picks (in the isochron groups used to determine the great circle and those being rotated) to the great circle ;

(iii) whether the target great circle is generated through fitting to 2 or 3+ picks,

$$w_{\mathrm{np}} = \begin{cases} 2 & n_{\mathrm{c}} = 2 \\ 1 & n_{\mathrm{c}} \geqslant 3 \end{cases} \tag{1.13}$$

because the azimuth of a great circle fitted through two points is going to be very sensitive to an error in one of the points, particularly when the two are very close together, and we would prefer the system to be biased towards great circles fitted to 3 or more picks;

(iv) the type of great circle to which the pick is being rotated (conjugate, central, non-conjugate),

$$w_{\mathrm{r}} = \begin{cases} 1 & \text{rotating to conjugate anomaly} \\ 1.5 & \text{rotating to ridge axis} \\ 2 & \text{rotating to independent anomaly} \end{cases} \tag{1.14}$$

because the most useful information is going to come from fitting anomalies to conjugate great circles, since this will not be affected by asymmetrical spreading or changes in ridge geometry. After this, picks rotated to/from the central anomaly are favourable because the rotation is dependent on only one rotation pole, and finally come picks rotated to a great circle of a different chron age.

The final weighting factor is a combination of these, except that all the picks with a weighting less than the standard deviation for the isochron data, σ^{mag}, are given unit weighting, and those with greater are assigned a normalised weight, such that the total weight W (including any bias,

β^{mag}) is given as

$$W = \begin{cases} \beta^{\mathrm{mag}} & (n_{\mathrm{gcs}} \times \xi \times w_{\mathrm{np}} \times w_r) \leqslant \sigma^{\mathrm{mag}} \\ \frac{\sigma^{\mathrm{mag}} \times \beta^{\mathrm{mag}}}{n_{\mathrm{gcs}} \times \xi \times w_{\mathrm{np}} \times w_{\mathrm{r}}} & (n_{\mathrm{gcs}} \times \xi \times w_{\mathrm{np}} \times w_{\mathrm{r}}) > \sigma^{\mathrm{mag}} \end{cases} \tag{1.15}$$

This may appear complex, but it is designed to reflect the fact that the most reliable source of data is expected to be a pick rotated to a great circle fitted to the conjugate anomaly group consisting of 3 or more picks.

1.2.2 Fracture zone data

The weighting for fracture zones is much simpler, and is based purely on the premise that the best sources of data are fracture zones which have the best overall fit to the synthetic flowlines. For a fracture zone with RMS error ξ, the weighting for each pick along the fracture zone is given by

$$W = \begin{cases} \beta^{\mathrm{fz}} & \xi \leqslant \sigma^{\mathrm{fz}} \\ \frac{\sigma^{\mathrm{fz}} \times \beta^{\mathrm{fz}}}{\xi} & \xi > \sigma^{\mathrm{fz}} \end{cases} \tag{1.16}$$

where σ^{fz} is the overall standard deviation for the fracture zone data, and β^{fz} the fracture zone biasing factor.

1.2.3 Transform fault data

As for the fracture zone data, the weighting for the transform data is simply a case of calculating a normalised weight for a particular transform relative to the overall distribution of errors for the transform data. So for a transform fault with RMS error ξ, the weighting for each pick along the transform is given by

$$W = \begin{cases} \beta^{\mathrm{tf}} & \xi \leqslant \sigma^{\mathrm{tf}} \\ \frac{\sigma^{\mathrm{tf}} \times \beta^{\mathrm{tf}}}{\xi} & \xi > \sigma^{\mathrm{tf}} \end{cases} \tag{1.17}$$

where σ^{tf} is the overall standard deviation for the transform fault data, and β^{tf} the transform biasing factor.

1.2.4 Outliers and data selection

An inevitable consequence of the data collection process is the presence of outliers. This is particularly so in the case of magnetics data, where the data need to be very carefully grouped. If a group of anomalies is included in a ridge section from which it should actually be separated, due to an unmapped fracture zone or offset of the ridge axis, then the errors for this group of data, or other picks rotated to this group (if it has a great circle fitted) will be spuriously high. This is because the errors arise not from any unsatisfactory pole parameters, but from an error in the grouping of the data itself. This applies equally for cases where there has been a change in ridge geometry or a past ridge jump, or where there has been asymmetric spreading.

Despite rigorous grouping, such outliers will still exist unless we choose to reduce the size of the dataset substantially. In some cases, in order to obtain groupings of picks big enough to generate data in some regions where anomaly picks are sparse, it is accepted that a certain number of outliers will be included (*e.g.* ensuring the inclusion of an isochron set with 2+ picks, in a group with other anomaly picks, so that a great circle can be fitted and used as a target for the other anomaly picks). This is sometimes necessary in order to have sufficient data points for the inversion to be meaningful, in the sense that poor data is better than no data at all. However, in order to reduce the effect of such outliers on the inversion, prior to the final inversion, data selection should be carried out, quite independent of any data weighting. A cutoff criterion Γ can be chosen such that any datum with a magnitude of misfit greater than Γ standard deviations will be discarded. The value of Γ is likely to be somewhat subjective, but a suitable value can be usually be determined after viewing q-q or histogram plots of the data misfits.

The problem of outliers is less important in the case of fracture zone and transform fault data. In particular it is very unlikely that there will be any significant outliers along a transform fault, and only a slight chance of a mis-pick caused by morphological anomalies along the transform valley, so Γ^{tf} can be set fairly high. For a fracture zone, there are two possible sources of outliers. The first is the whole fracture zone existing as an unrepresentative flowline, possibly due to plate deformation warping the trace of the fracture zone, or asymmetric spreading which would lead to a fracture zone deviating from a synthetic flowline. Additionally there is the possibility of errors introduced during the picking procedure. However, general confidence in the fracture zone picks is substantially higher than for the magnetics data, and so it is possible to set Γ^{fz} correspondingly higher if desired.

1.3 Statistical analysis

1.3.1 Distribution of errors

A key factor of the inversion procedure is an accurate estimation of errors. The initial assumption is made that there is a Gaussian distribution, though it is accepted that this assumption will become invalid towards the 'tails' of the distribution, with the inevitable presence of outliers in the dataset. A reliable test of the theoretical Gaussian distribution of the errors in the data is made by carrying out q-q (quantile-quantile) analysis.

This procedure consists of plotting the quantiles of the distribution of calculated error values (misfits) against the corresponding quantiles of the theoretical distribution (usually a Gaussian distribution of mean 0 and standard deviation 1). If the two distributions are identical, then the quantiles would also be defined equally, and so a simple straight line (y=x) would result.

Testing against a (0,1) Gaussian distribution has the advantage that if the distribution is Gaussian, then the plot will be linear with the zero intercept yielding an estimate of the mean of the sample dataset, and the gradient an estimate of the standard deviation. Because of the nature of the datasets, the q-q plots are expected to be linear close to the origin, but deviating from the straight line when outliers become apparent.

The q-q plots are a very good visual test for the Gaussian nature of the distribution of errors, with reliable estimates of the sample statistics able to be determined. Through studying where the plots deviate from a straight line, it is easy to spot where the distribution of errors ceases to be Gaussian (indicating the presence of outliers) and hence identify suitable values for Γ.

1.3.2 Confidence intervals

The confidence intervals used in this study are those obtained from the covariance matrix which is intrinsically defined by the inversion.

The governing equation for the inversion, given by (1.8), is

$$\hat{\mathbf{A}}\,\Delta\mathbf{p} = \hat{\mathbf{b}} \tag{1.18}$$

where $\hat{\mathbf{A}}_{ij} = \frac{\partial \hat{\varepsilon}_i}{\partial p_j}$ is the partial derivative matrix showing how changes in the pole parameters affect the normalised errors $\hat{\varepsilon}$. From this, the matrix

$$\left[\hat{\mathbf{A}}^{\mathrm{T}}\hat{\mathbf{A}}\right]_{ij} = \frac{\partial \hat{\varepsilon}_k}{\partial p_i}\frac{\partial \hat{\varepsilon}_k}{\partial p_j} \tag{1.19}$$

is the correlation matrix for changes in the errors of the data with respect to the poles, and then the inverse,

$$\mathbf{C} = \left[\hat{\mathbf{A}}^{\mathrm{T}}\hat{\mathbf{A}}\right]^{-1} \tag{1.20}$$

is the auto-covariance matrix for the solution, relating uncertainties in the solution poles to the uncertainties in the data.

This is easily seen by considering that if the solution has a high uncertainty, then another parameter set close to the determined solution set $\tilde{\mathbf{p}}$ would not alter the errors significantly, hence the values of $\hat{\mathbf{A}}^{\mathrm{T}}\hat{\mathbf{A}}$ would be low, leading to large values for $\mathbf{C}$. The validity of this as a method of determination of the covariance matrix relies strongly on the assumption that the problem is linear in the region of the solution. It is important to note that the error vector $\mathbf{b}$ does not appear, this covariance matrix is generated by the relative stabilities of the errors, as opposed to the absolute errors themselves.

The normalised matrix $\hat{\mathbf{A}}$ is used, before applying the pick weights, as this is the purest equation of the inversion and gives a true measure of how changes in the pole parameters will affect the errors, ahead of any bias that we might wish to give the inversion, as created by the weighting process. If we consider the effects of including the weights and data biases, from equations (1.11) and (1.20), we see that by weighting down a large amount of data, the values of the covariances would increase, or similarly by having no weighting, but biasing the values in favour of one data type the covariances would decrease. However, this does also show the importance of determining the correct distribution of errors for each data type, since an error in the standard deviations used to normalise the data would lead to a corresponding error in the covariance matrices, and hence the confidence ellipses.

From the matrix **C**, the 3x3 submatrices for the individual poles can be extracted, such that the symmetric covariance matrix $\mathbf{C}^i$ for the i'th pole $(\theta^i, \phi^i, \alpha^i)$ is obtained, The next step is to determine the eigenvectors of this matrix, which will represent the semimajor axes of the confidence ellipse in 3-D parameter space, and the corresponding eigenvalues which yield the squared variances along these axes. The confidence ellipsoid is defined by these three axes, and the magnitude is determined by referring to χ^2 tables, such that the axes are multiplied by a factor of 2.79 to represent 95% confidence in three dimensions. Sometimes it is easier to represent the confidence intervals as an ellipse rather than a surface, where the angle is considered fixed. In this case the 2x2 submatrix for (θ^i, ϕ^i) is used, and from the eigenvalues and eigenvectors of this, the 95% confidence region for 2 degrees of freedom is represented by an ellipse of 2.45σ.

1.3.3 Data importances

Whilst running inversions, a useful property to know is the data importance of each pick, which is a measure of how important each datum is to the final inversion. The value of importance for the i'th datum in a dataset is defined as $\mathcal{I}_i$, the i'th element on the diagonal of the symmetric matrix **P** where **P** is defined as

$$\mathbf{P} = \mathbf{A}(\mathbf{A}^\mathsf{T}\mathbf{A})^{-1}\mathbf{A}^\mathsf{T} \tag{1.21}$$

where **A** is the partial derivative matrix for the dataset. This can either be the matrix for the whole dataset, or a subset such as the partial derivative matrix for the fracture zone picks alone.

Since only the elements along the diagonal of **P** are required, the calculation can be simplified by avoiding building the large matrix **P** by defining

$$\mathcal{I}_i = \mathbf{A}_i(\mathbf{A}^\mathsf{T}\mathbf{A})^{-1}\mathbf{A}_i^\mathsf{T} \tag{1.22}$$

where $\mathbf{A}_i$ is the i'th row of the matrix **A**. Each value of importance is a positive number, and it is a measure of how the changes in the i'th datum correlate with the stability of the model. If $\mathcal{I}_i$ is large, then this suggests that the uncertainties in the i'th datum are linked to those in the model, and so this datum has a significant effect on the inversion. Similarly the converse applies, if a datum has a low importance then it will have less effect on the result, and is relatively redundant. In other words, the importance is a measure of the contribution of the i'th datum to the model relative to the dataset in general, and thus is a measure of this point's independence from this dataset. As with the covariance matrix, the actual errors (**b**) do not appear in the determination of the importances, which are generated by the nature and distribution of the dataset, and the distribution of information within the dataset.

Large values of $\mathcal{I}_i$ are bad for the inversion, since if there are any inherent errors in the data with large importances, then this will be carried through significantly into the inversion. Ideally all the importances would be equal, suggesting a representative distribution of data with no single datum having a significant effect on the result. Furthermore, considering the importance as the measure of the redundancy of that datum, a large value would tend to suggest that similar data, of that type, or in that region, would be desirable.

The importance values are additive, and so the importance of a subset of data can be obtained by summing the importances of the subset. This is particularly relevant for the magnetics dataset, as one pick may yield several data (Figure 1.2), and so the overall importance of that pick is simply found by adding the importances of each datum generated from that pick. As a rule, the data importances are usually determined for the different types of data separately, as this shows clearly where there are areas with sparse data coverage. The data importances should always be determined from **A**, the final normalised and weighted matrix, as this represents the true effect of each data point on the result of the inversion.

1.4 Extension to a 3-plate problem

1.4.1 General consequences

When the problem is extended to three plates (1, 2 and 3), there is a set of l finite rotation poles (and hence $n = 3l$ pole parameters) for each plate pair, leading to a total of $3n(9l)$ pole parameters. However, because of the nature of the plate circuit, if we know the poles for two of the plate pairs, then we already have all the information we need to determine the third set of poles, so, we only in fact have $2n$ independent pole parameters. We take the first two sets of poles, $1 \mapsto 2$ and $2 \mapsto 3$, and use the parameters for these two sets of poles as the parameter set for the three plate inversion.

For the first plate pair, the calculations are carried out as before, the misfits determined, and the partial derivatives calculated relative to pole parameters $\{p_1, ..., p_n\}$. For the second plate pair, again the calculations are carried out as for a two plate problem, but with the partial derivatives being calculated and stored relative to pole parameters $\{p_{n+1}, ..., p_{2n}\}$. For the third set of data, $1 \mapsto 3$, a third set of finite rotation poles is calculated and the misfits determined from this set of poles, as for a 2-plate inversion. However, when calculating partial derivatives, the effect of this third set of poles is ignored and instead the derivatives are determined with respect to the two sets of poles which were used to obtain the third set.

The final partial derivative matrix is constructed from the three partial derivative matrices from each plate pair, but all entries in the **A** matrix will be zero for parameters $\{p_{n+1}, \cdots, p_{2n}\}$ for data from the 1↦2 dataset or for $\{p_1, \ldots, p_n\}$ from the 2↦3 dataset. Thus, the matrix $\mathbf{A}$ is given by

$$\mathbf{A} = \begin{bmatrix} {}^{12}\mathbf{A}_{1,1} & \ldots & {}^{12}\mathbf{A}_{i,n} & 0 & \ldots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ {}^{12}\mathbf{A}_{{}^{12}m,1} & \ldots & {}^{12}\mathbf{A}_{{}^{12}m,n} & 0 & \ldots & 0 \\ 0 & \ldots & 0 & {}^{23}\mathbf{A}_{1,n+1} & \ldots & {}^{23}\mathbf{A}_{1,2n} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & \ldots & 0 & {}^{23}\mathbf{A}_{{}^{23}m,n+1} & \ldots & {}^{23}\mathbf{A}_{{}^{23}m,2n} \\ {}^{13}\mathbf{A}_{1,1} & \cdots & \cdots & \cdots & \cdots & {}^{13}\mathbf{A}_{1,2n} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ {}^{13}\mathbf{A}_{{}^{13}m,1} & \cdots & \cdots & \cdots & \cdots & {}^{13}\mathbf{A}_{{}^{13}m,2n} \end{bmatrix} \tag{1.23}$$

where a prefix such as ^{12} denotes a quantity or parameter for the plate pair 1↦2.

The basic theory behind the inversion remains unchanged, there is just a larger amount of data, and double the number of pole parameters. The inversion will now be solving for a parameter set consisting of the two subsets of pole parameters ${}^{12}\mathbf{p}$ and ${}^{23}\mathbf{p}$, whilst in the process effectively also solving for the third set, in order to satisfy equally the datasets on all three plates. The output of the inversion will be adjustments to the first two sets of pole parameters, and when these new poles have been calculated, the new set of poles for the third plate can be determined. It makes no difference in which order the plate circuit is used or which set of poles is considered to be the redundant set, the same solution should be reached, within the limits of operational error (this is checked during the inversion process).

1.4.2 Partial derivatives

Where in a 2-plate inversion for the plate pair 1↦3 we would determine partial derivatives with respect to the parameters of the k'th rotation pole ${}^{13}\mathbf{F}^k$, $\left({}^{13}\theta^k, {}^{13}\phi^k, {}^{13}\alpha^k\right)$, in a 3-plate inversion, partial derivatives are instead determined for the pole parameters of the two independent poles used to determine ${}^{13}\mathbf{F}^k$, which are $\left({}^{12}\theta^k, {}^{12}\phi^k, {}^{12}\alpha^k, {}^{23}\theta^k, {}^{23}\phi^k, {}^{23}\alpha^k\right)$. This makes the calculation of partial derivatives a far harder task, but this isn't covered here. For more details read the bible (Adrian Nankivell's D.Phil Thesis - University of Oxford, 1997).

1.4.3 Statistical analysis

With the extension of the method to 3 plates, there is no change in the overall inversion theory as detailed before, just an increase in the number of parameters. The distributions of the data are still considered to be Gaussian and can be tested using q-q plots. The data importances are calculated just as before, from equation (1.22), with no further steps needing to be taken.

However, in order to determine the confidence intervals for the poles, a little more care needs to be taken. Equation (1.20) will determine the autocovariance matrix for the inversion, as with the 2-plate method. From this can be obtained the 3x3 submatrices that describe the covariances of each set of 3 pole parameters that are determined by the inversion. This is only possible for the first two sets of poles however, as the inversion was only solving for these parameters. Intuitively, there must be confidence intervals that exist for the third set of poles - if the other poles were known exactly, then of course, the third set would be defined exactly, but while there are uncertainties in the location of the first two sets, there will be corresponding uncertainties in the locations of the third set.

So, considering that it should make no difference to the result which direction the plate circuit is taken, or which two sets of parameters are used, it follows that the confidence ellipses for each set of poles will be unchanged by the choice of plate pairs. If initially inverting for $1 \mapsto 2$ and $2 \mapsto 3$, the covariance matrix ${}^{23}\mathbf{C}$ should be identical (again, within operational error) to that obtained from inverting for $2 \mapsto 3$ and $3 \mapsto 1$. And so the covariance matrix ${}^{31}\mathbf{C}$ obtained from this second inversion should also be valid. The procedure is then to determine a solution $\tilde{\mathbf{p}} = \{{}^{12}\tilde{\mathbf{p}}, {}^{23}\tilde{\mathbf{p}}\}$ for the circuit $1 \mapsto 2 \mapsto 3$, and from the partial derivative matrix $\mathbf{A}(\tilde{\mathbf{p}})$, the covariance matrices ${}^{12}\mathbf{C}$ and ${}^{23}\mathbf{C}$ are obtained. The best-fit parameter set for $1 \mapsto 3$, ${}^{13}\tilde{\mathbf{p}}$, is then determined from ${}^{12}\tilde{\mathbf{p}}$ and ${}^{23}\tilde{\mathbf{p}}$, and the circuit is then changed to $1 \mapsto 3 \mapsto 2$. The partial derivative matrix $\mathbf{A}(\tilde{\mathbf{p}}) = \mathbf{A}({}^{13}\tilde{\mathbf{p}}, {}^{32}\tilde{\mathbf{p}})$ is then calculated and from this, the covariance matrix ${}^{13}\mathbf{C}$ determined. (The parameter set ${}^{32}\tilde{\mathbf{p}}$ is the same as ${}^{23}\tilde{\mathbf{p}}$, except that the signs of the rotation angles are reversed).

2

Preparation of input data

The reconstruction programs take three types of input data: magnetic anomaly reversal picks; transform fault picks; and fracture zone picks.

2.1 Fracture zone picks

For each fracture zone, a seed point needs to be obtained, from which to generate the synthetic flowline to which the fracture zone picks will be fitted. The best possible seed point is naturally the ridge-transform intersection for that fracture zone, corresponding to the zero-age point of the fracture zone and flowline. Sometimes this is not possible where a fracture zone was created at a spreading centre no longer in existence. In other cases, the transform faults which gave rise to some fracture zones no longer exist, as a result of re-organisation and changes in segmentation on the ridge itself. Consequently, the locations of the corresponding ridge-transform intersections are sometimes not easily estimated, and so it would be highly speculative to attempt to determine a point along the ridge from where the fracture zone trace might have been generated. In these cases, a seed point needs to be located somewhere along the fracture zone, at an age corresponding to one of the anomaly chrons to be used in the reconstruction. In this way, a non-zero-age seed point can be used just as easily to generate a flowline for the current poles.

Seed point locations for the ridge-transform intersections are best identified from satellite gravity maps, where they are often clearly visible. Alternatively, if the fracture zone begins in an area where there are several magnetic anomaly identifications for the central Brunhes anomaly, these can be used to determine the zero-age point. For fracture zones where it is not possible to pick a zero-age seed point, then the best case scenario is to use an isochron pick on or close to the fracture zone to determine a seed point. If the nearest isochron picks are not very close to the fracture zone, then it is possible to interpolate a theoretical isochron from nearby picks (taking the local direction of spreading into consideration) in order to determine an intersection with the fracture zone. For these methods, it is important to ensure that anomalies on the younger flank of the fracture zone were used (*c.f.* ridge-transform intersection).

In other cases, if the fracture zone has a distinctive feature, such as a kink at a change of spreading direction, which is mirrored in a neighbouring fracture zone which has sufficient isochron picks in the proximity to be able to confidently determine an age for that feature, then this dating can be used to provide a seed point for the undated fracture zone at that feature. Some of these methods are somewhat subjective in the choice of seed and often the choice has to be made to exclude a fracture zone from the dataset if a satisfactory seed point cannot easily be determined.

Any mistakes in setting a seed point soon become apparent as an inversion proceeds, and are easily rectified. In cases where there is a significant change in spreading direction, fracture zones which are badly dated give rise to flowlines which clearly have a kink, or altered direction, in an obviously incorrect place, giving rise to substantial mismatches between the fracture zone and the synthetic flowline. This is also often indicated by a large shift in the seed point, where the fracture zone processing routine is attempting to find a best-fit flowline for the fracture zone, and failing to reach a satisfactory solution within the allowed limits. If this occurrs, then an attempt should be made to find a more consistent seed for the fracture zone, but if this is not possible and the choice of the seed becomes too subjective, then that fracture zone should be eliminated from the fracture zone dataset. There are two other reasons for excluding fracture zones from the main dataset: if a fracture zone section is particularly short in comparison to the rest in the dataset; or if there were a large number of fracture zones in a given region where the importance of the data is determined to be minimal.

The fracture zone dataset comprises of two parts: a series of files containing the seed information and picks for a single fracture zone; and a 'master' file listing all of the above files.

An individual fz file gives the 'seed' information first - the first line giving the lat lon location of the seed point to be used, then a second line giving the plate id for the fracture zone, and then an anomaly id for the seed (a3,x,a4). This anomaly id can left out if it the reference/seed point is the ridge-transform intersection (*i.e.* c1), but otherwise must be one of the chrons for which the inversion is being carried out. After this seed information follows a list of lat, lon locations for picks along the fracture zone (No special character is required at the end of this file). So a single fracture zone file appears as follows:-

```
-51.744186 36.584831
ant  c5
-51.168900 37.047401
-51.308102 36.839600
-51.483501 36.762501
-51.666199 36.711399
.....ctd.
```

The main FZ data file begins with the number of fracture zones to be used, followed by a list, in no particular order, of the individual fracture zone data filenames/locations. The file locations should be with respect to the directory from which the inversions are to be run from, so that a list file would appear as:

```
60
afrant/fz/fz10n_recon.dat
```

```
afrant/fz/fz10nb_recon.dat.1
afrant/fz/fz10s_recon.dat
afrant/fz/fz11n_recon.dat
afrant/fz/fz11s_recon.dat.1
afrant/fz/fz12n_recon.dat.1
afrant/fz/fz12s_recon.dat.3
afrant/fz/fz13n_recon.dat.2
afrant/fz/fz13s_recon.dat.1
afrant/fz/fz14s_recon.dat
afrant/fz/fz17n_recon.dat
.....ctd.
```

2.2 Magnetics anomaly picks

After ensuring in the initial processing stages that an internally consistent set of anomaly picks has been obtained (*i.e.* no immediately obvious outliers), the important step is to split the dataset up into groups that were created at the same ridge segment. This can be done by studying the fracture zones and splitting up any anomaly groups that are clearly separated by a large offset fracture zone. In many cases, in a region where there is a large set of anomalies of a particular age, any offset in the groups is usually clearly visible and so the division is simple. Because of the nature of the reconstruction routines, it is possible to make an anomaly grouping which only contains picks on one side of the ridge and yet not preclude the use of these picks in the inversion. However, it is preferable for a group to contain picks on both sides of the ridge in order to be able to carry out as much conjugate fitting as possible, and so one-sided groups should only be constructed when this is unavoidable, usually where only one side of the ridge is in existence, due to past subduction processes.

The magnetic isochron picks need to be preprocessed before entry into the inversion routines, in order to determine great circles, and allocate target great circles for each set of picks.

The format of a raw magnetics data file is given below. A line containing information about a magnetics pick is formatted as follows:

columns 1-4: anomaly id (character string, a4)
columns 5: (space, x)
columns 6-8: plate id (a3)
columns 9-10: (x)
columns 11-20: latitude (f10.4)
columns 21-30: longitude (f10.4)
columns 31: (x)
columns 32-37: pick id (integer,i6)

so that a single line looks like

```
  c5 sam     -9.7140  -15.0098 702008
```

Each set of picks designated as being created within the same ridge section must be separated by xxxx, and though not required, each set of picks for the same isochron can be separated by ----. Finally, the EOF is marked by .... , so a complete set of picks might look like:

```
  c1 ant    -54.2238    3.9126 220001
  c1 ant    -54.3752    4.2973 221003
  c1 ant    -54.0166    3.4264 201003
----
 c2A ant    -54.0109    3.2226 219007
 c2A ant    -54.1561    3.4710 219008
 c2A ant    -54.3267    3.7519 220002
----
  c5 ant    -54.6736    2.9500 235002
  c5 ant    -54.4647    2.5881 219014
----
 c2A afr    -53.8362    3.8570 219009
 c2A afr    -53.9669    4.0777 219010
 c2A afr    -53.8424    3.9152 219011
 c2A afr    -54.0386    4.0063 219012
----
  c5 afr    -53.2753    4.5685 212007
  c5 afr    -53.6771    5.5137 219005
  c5 afr    -53.5690    4.8579 220004
----
  c6 afr    -53.0761    5.4359 220005
xxxx
  c5 ant    -52.6438   13.2361 231026
----
  c1 ant    -52.2346   14.5292 231001
----
 c2A ant    -52.2719   14.4083 231019
 c2A ant    -52.3591   14.4875 231023
----
 c2A afr    -51.9788   14.4833 231018
 c2A afr    -52.0662   14.8708 231020
----
  c5 afr    -51.5599   14.9972 231021
  c5 afr    -51.4818   14.7139 231024
....
```

This raw datafile must then be run through the preprocessing program **magprepro**. The input to this program is the raw datafile (**magex.raw** in the following example) and also the set of initial 'test' poles (**poles_01_14.dat**), which are used to determine which anomalies are being inverted for and therefore can be used as target great circles. This program allows three different modes of fitting and matching great circles. The first option only fits great circles to groups of three or more picks, and attempts to allocate conjugate great circles to anomaly group within the ridge section. In the case where there is no conjugate great circle for a set of picks, then they are allocated all of the fitted great circles within the ridge section as 'target' great circles. The second option considers the case where a ridge section contains many picks, but there is no group of picks containing more than two picks. In this case, the exact great circle joining each pair of picks is calculated, and these are listed. The user is then asked which, if any, anomaly pair should be used as the sole target great circle for all of the picks within the ridge section. The final option is to only allow fitting to conjugate anomalies, in which case great circles are only fitted to groups of three or more picks, and any groups of picks which cannot be fitted to a conjugate great circle are excluded from the output dataset.

The processing of the raw dataset shown above is given below, with user input shown in bold. The 'fitting to selected 2-pick groups' processing option is chosen.

```
% magprepro
 filename with picks to be sorted:
 magex.raw
 opening magex.raw   for read.
 relevant poles filename:
 data/afrant/poles_01_14.dat
 opening data/afrant/poles_01_14.dat  for read.
 output filename:
 magex.out
 opening magex.out  for write.
 gc plot data filename:
 magex.gcplot
 opening magex.gcplot   for write.
 gc pick data filename:
 magex.gcpicks
 opening magex.gcpicks for write.

  Fit GCs to only 3-pick groups           - enter 3
       also to a selected 2-pick group    - enter 2
  and rotate single picks to each other  - enter 1
  only rotate to conjugate anomalies     - enter 0
 2
Ridge Section:-   1
```

```
Anomaly Group ant   c1   3 picks - Great Circle fitted
Anomaly Group ant  c2A   3 picks - Great Circle fitted
Anomaly Group ant   c5   2 picks - EXACT Great Circle calculated
Anomaly Group afr  c2A   4 picks - Great Circle fitted
Anomaly Group afr   c5   3 picks - Great Circle fitted
Anomaly Group afr   c6   1 picks
end of ridge section -   6 anomaly groups
Rotating   c1 ant picks to  c2A ant great circle
Rotating   c1 ant picks to  c2A afr great circle
Rotating   c1 ant picks to   c5 afr great circle
Rotating  c2A ant picks to  c2A afr great circle
Rotating   c5 ant picks to   c5 afr great circle
Rotating  c2A afr picks to  c2A ant great circle
Rotating   c5 afr picks to   c1 ant great circle
Rotating   c5 afr picks to  c2A ant great circle
Rotating   c5 afr picks to  c2A afr great circle
Rotating   c6 afr picks to   c1 ant great circle
Rotating   c6 afr picks to  c2A ant great circle
Rotating   c6 afr picks to  c2A afr great circle
Rotating   c6 afr picks to   c5 afr great circle
xxxx - NEXT RIDGE SEGMENT

Ridge Section:-   2

Anomaly Group ant   c1   1 picks
Anomaly Group ant  c2A   2 picks - EXACT Great Circle calculated
Anomaly Group ant   c5   1 picks
Anomaly Group afr  c2A   2 picks - EXACT Great Circle calculated
Anomaly Group afr   c5   2 picks - EXACT Great Circle calculated
end of ridge section -   5 anomaly groups
Section   2 $>$ Max. of 2 picks in a group
Select number of anomaly group to fit GC to from list below
 0 : fit none
 1 : ant  c2A
 2 : afr  c2A
 3 : afr   c5
 1
Rotating   c1 ant picks to  c2A ant great circle
no target for anom set ant  c2A
Rotating   c5 ant picks to  c2A ant great circle
Rotating  c2A afr picks to  c2A ant great circle
Rotating   c5 afr picks to  c2A ant great circle
```

```
STOP: ok
```

The output file can then be used directly in the inversion routines, and looks as follows:-

```
  c1 ant       3  3
  -54.2238    3.9126 220001
  -54.3752    4.2973 221003
  -54.0166    3.4264 201003
 c2A ant     24.1967   55.0021  0.14  0.20     3
 c2A afr     15.6498   71.3532  2.99  4.28     4
  c5 afr     29.2312   45.8251  7.19 10.15     3
-----------------------------------------------
 c2A ant       3  1
  -54.0109    3.2226 219007
  -54.1561    3.4710 219008
  -54.3267    3.7519 220002
 c2A afr     15.6498   71.3532  2.99  4.28     4
-----------------------------------------------
  c5 ant       2  1
  -54.6736    2.9500 235002
  -54.4647    2.5881 219014
  c5 afr     29.2312   45.8251  7.19 10.15     3
-----------------------------------------------
 c2A afr       4  1
  -53.8362    3.8570 219009
  -53.9669    4.0777 219010
  -53.8424    3.9152 219011
  -54.0386    4.0063 219012
 c2A ant     24.1967   55.0021  0.14  0.20     3
-----------------------------------------------
  c5 afr       3  3
  -53.2753    4.5685 212007
  -53.6771    5.5137 219005
  -53.5690    4.8579 220004
  c1 ant     28.5647   44.8509  0.23  0.33     3
 c2A ant     24.1967   55.0021  0.14  0.20     3
 c2A afr     15.6498   71.3532  2.99  4.28     4
-----------------------------------------------
  c6 afr       1  4
  -53.0761    5.4359 220005
  c1 ant     28.5647   44.8509  0.23  0.33     3
 c2A ant     24.1967   55.0021  0.14  0.20     3
 c2A afr     15.6498   71.3532  2.99  4.28     4
  c5 afr     29.2312   45.8251  7.19 10.15     3
------------------------------------------------
  c1 ant       1  1
  -52.2346   14.5292 231001
 c2A ant     17.2623   80.7271  0.00  0.00     2
------------------------------------------------
  c5 ant       1  1
  -52.6438   13.2361 231026
 c2A ant     17.2623   80.7271  0.00  0.00     2
------------------------------------------------
 c2A afr       2  1
  -51.9788   14.4833 231018
  -52.0662   14.8708 231020
 c2A ant     17.2623   80.7271  0.00  0.00     2
------------------------------------------------
  c5 afr       2  1
  -51.5599   14.9972 231021
  -51.4818   14.7139 231024
 c2A ant     17.2623   80.7271  0.00  0.00     2
------------------------------------------------
```

The first line of each section gives the anomaly id and plate id for a set of picks, followed by the number of picks in the group, and the number of great circles to which the picks are to be rotated to (a4, x, a3, 2x, i6, x, i2). The next line(s) contain the actual lat lon location(s) and id(s) (2f10.4, x, i6) for as many picks as there are in the group. After this come the details of the target great circle(s), one line for each, containing the anomaly id, plate id, great circle pole location (lat lon), the RMS error in km for the picks to which the great circle was fitted, the maximum error, and the number of picks which were used to generate the great circle (a4, x, a3, 2x, 2f10.4, 2f6.2, i6).

There are two other output files, which can be used to display features of the great circles fitted:

magex.gcplot contains, for each great circle used, two points which lie on the great circle and describe the arc of the great circle which contains all of the closest points of the picks to which the great circle was fitted. Thus for where an exact great circle was calculated for a pair of anomaly picks, these plot points will be the original two pick points.

magex.gcpicks contains series of xy pairs, separated by > markers, for each set of picks to which a great circle was fitted.

2.3 Transform fault picks

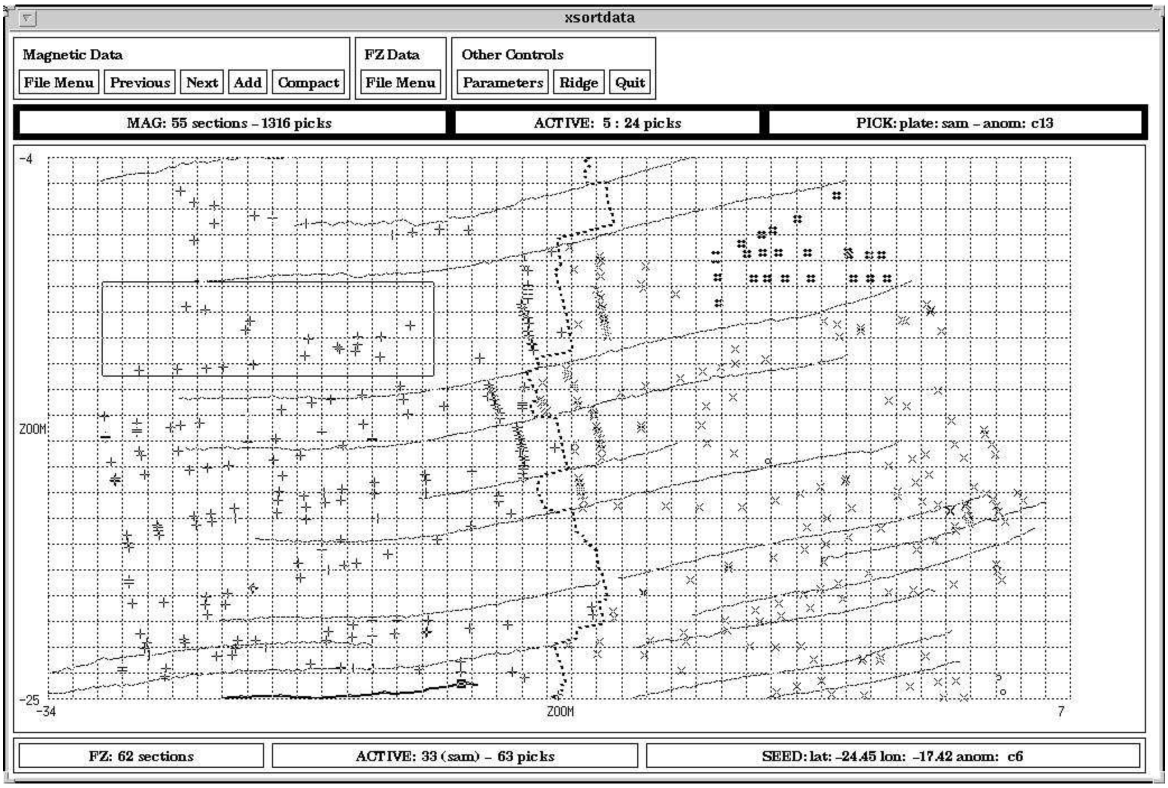
The format for transform fault picks is much simpler - it it a lat lon list of picks, with each transform section separated by a > marker, so that a short file containing only two transform sections would appear like:-

```
-54.8218 1.1808
-54.7100 1.4270
-54.5848 1.6501
-54.4528 1.8611
-54.3268 2.0815
-54.1876 2.2789
-54.0481 2.4750
-53.9337 2.7130
-53.8251 2.9606
-53.6971 3.1743
-53.5860 3.4008
>
-54.7359 5.4812
-54.5812 5.6515
-54.4634 5.8877
-54.3205 6.0777
-54.1895 6.2888
-54.0521 6.4874
-53.9388 6.7294
-53.8068 6.9365
-53.6436 7.1898
>
```

2.4 xsortdata

Once the basic picks have been made for both sets of data, *i.e.* sets of picks for a number of fracture zones and a large number of anomaly picks, then it is often desirable to view the two sets of data together to enable the final preparation stages, detailed above, to be carried out. This enables fracture zones to be 'dated' by viewing any isochron picks in the vicinity, and to group anomalies by studying whether any given group is bisected by a mapped fracture zone. To make these tasks easier a further program was written for X-Windows, **xsortdata**. A screenshot of this program is shown in Figure 2.1.

All the magnetic picks are loaded as a single raw datafile, through use of the `Magnetic Data` - `File Menu` - `Load` buttons. and the fracture zone data loaded by initially reading in the main list file (`FZ Data` - `File Menu` - `Load` buttons), and then the individual fracture zone data files are all loaded in turn. Both these sets of data are then displayed simultaneously on screen, set onto a flat-earth basemap. The magnetic picks are displayed as crosses, 'x' or '+', depending on which side of the ridge they are located, and the fracture zones displayed as lines linking the picks made for that fracture zone. The boundaries of the display can be set, and also the plate id's for



MAG: 55 sections – 1316 picks

ACTIVE: 5 : 24 picks

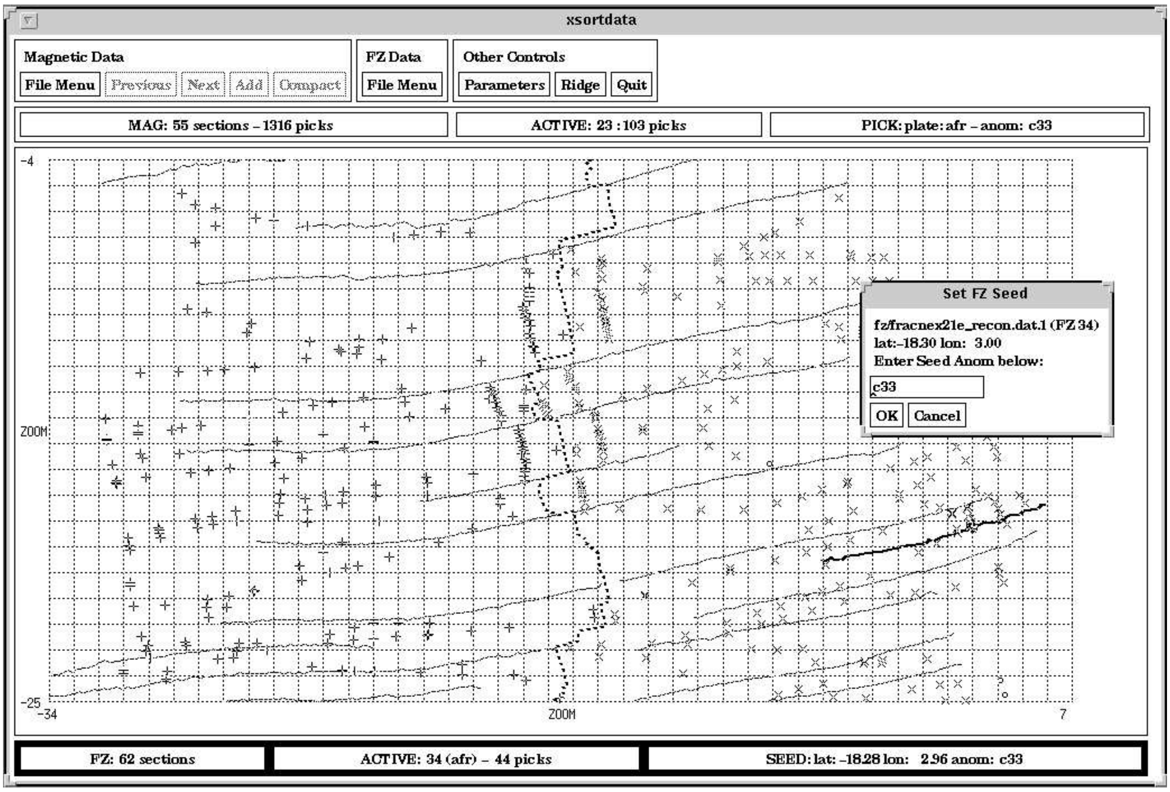
PICK: plate: sam – anom: c13

FZ: 62 sections

ACTIVE: 33 (sam) – 63 picks

SEED: lat: -24.45 lon: -17.42 anom: c6

(a) Selecting a group of isochron picks.



MAG: 55 sections – 1316 picks

ACTIVE: 23 : 103 picks

PICK: plate: afr – anom: c33

fz/fracnex21e_recon.dat.1 (FZ 34)
lat: -18.30 lon: 3.00
Enter Seed Anom below:
c33

FZ: 62 sections

ACTIVE: 34 (afr) – 44 picks

SEED: lat: -18.28 lon: 2.96 anom: c33

(b) Setting the reference 'seed' point for a fracture zone.

Figure 2.1: The interactive plate reconstruction data processing program **xsortdata** in use.

the ‘x’ and ‘+’ picks set using the `Parameters` button. The current ridge axis can also be loaded (as an x,y file with > seperators) using the `Ridge` button, and is displayed as a thick dashed line.

The map display can also be altered using the following mouse and keyboard operations:

↑ Shift + ▮▯▯ Zooms in to box defined by hold and drag operation.

↑ Shift + ▯▮▯ Zooms out to full map display.

↑ Shift + ▯▯▮ Redraws current map display.

Both sets of data can be edited and saved independently, and additionally either dataset can be cleared at any stage using the `File Menu` - `Clear` buttons. The program only allows operations to be carried out on one datatype at one time, and control between the two is changed by clicking on the magnetics and fracture zone information/display bars, located just above and below the main map respectively. The information/display bar for the dataset which is currently being edited has a thick black border around it.

When editing the magnetics dataset, the picks can be split up into multiple ridge sections, containing as few or as many picks as required. The `Magnetics Data` - `Previous` and `Next` buttons can be used to cycle through the current groups of picks, and the `Add` button adds an additional (empty) set. Because during the transfer of some picks from one group to another may cause some groups to become empty, the `Compact` button deletes all empty sets, and renumbers the groups accordingly.

Picks can only be added to the group which is shown to be active in the magnetics information/display bar, and all the picks in this ridge section are highlighted (**+** and **x** as opposed to + x). The keyboard and mouse operations in ‘magnetics’ mode are as follows:

▮▯▯ or ▯▮▯ or ▯▯▮ Display pick information (when clicking on any displayed pick). First click gives plate and anomaly id’s, the second gives latitude and longitude and third gives pick id.

Control + ▮▯▯ Transfers nearest pick (click) or group of picks within box (hold and drag) to active group as picks on plate 1 (+).

Control + ▯▮▯ Transfers nearest pick (click) or group of picks within box (hold and drag) to active group as picks on plate 2 (x).

Control + ▯▯▮ Undoes all changes to nearest pick (click) or group of picks within box (hold and drag) since active group was last changed.

↑ Shift + Control + ▯▯▮ Toggles an ‘unreliable’ pick. Clicking on any pick, whether in the active group or not, will cause the pick to be considered ‘dead’ and displayed as a square. Clicking on the square will return the pick to the dataset, in the **active** group. Picks marked as dead will only be written in the output file after the .... EOF marker and so will be ignored by the preprocessing routine. However, they will appear, as ‘dead’ picks, if this output file is later reloaded.

When all the desired editing of the magnetic picks has been carried out, the data can then be saved as a single file using the `Magnetic Data` - `File Menu` - `Save` buttons. In this file, the magnetics picks are split up into the ridge sections as chosen interactively and, within these, into anomaly groups and is formatted for direct entry in the preprocessing stages of the inversion procedure. After studying the results of an inversion, the magnetics dataset can be directly re-entered into this routine to rearrange the groupings, sub-divide ridge sections further, or remove any obvious outliers.

When editing the fracture zone data, only one fracture zone is ‘active’ at any stage, and is drawn as a line twice the thickness of the other fracture zones, and the seed point for that fracture zone marked as a square. The keyboard and mouse operations in ‘fracture zone’ mode are as follows:

▮▯▯ Change active fracture zone. Details of the fracture zone are shown on the fracture zone information/display bar. A second click will display the filename of the fracture zone.

Control + ▮▯▯ Toggle status of active fracture zone. The fracture zone can be removed or returned to the main dataset. When chosen as undesirable, then the fracture zone will displayed as a dotted line (except when ‘active’).

Control + ▯▮▯ Set new seed for active fracture zone. A box appears giving details of the fracture zone and the location at which the mouse button was clicked and requests an anomaly id for the new seed.

Control + ▯▯▮ Edit pick locations along active fracture zone. When these buttons are held down, a small box will appear at the location of the pick along the fracture zone which is closest to the cursor. If the crosshairs of the cursor are within this box when the mouse button is released, then this pick will be removed from the active fracture zone (which will be re-drawn accordingly). This pick will now be drawn as a much larger box when the same combination of keyboard and mouse buttons are held down again, but can be returned to the fracture zone by following the same procedure as before. Any picks removed from a fracture zone in this way will not be saved at a later date.

When all the required sorting is completed, the edited fracture zone dataset can be saved using the `FZ Data` - `File Menu` - `Save` buttons. For each fracture zone for which details have been changed, a new file is written, or the old file over-written depending on the status of the `Overwrite - YES/NO` button. A single file will then be written listing the filenames for all the chosen fracture zones. All are in the exact format required for the inversion routines, and can be used without any further alteration. As such, these files can be continually re-entered into the program to adjust any obviously incorrect seed points, or to remove individual picks within a fracture zone that are giving rise to large errors, or to remove whole fracture zones from the inversion procedure.

3

2-plate reconstructions

Once the datasets were in a suitable format for entry into the main inversion procedure, the files were transferred into a single working directory.

3.1 Required input files

The iteration file must be called **iter.dat** and the iteration number must be in columns 21-22, as in the following example:

```
 Iteration number =  1
```

The inversion parameter control file must be called **inv.dat** and contains the information required to drive the least squares routine and also important weighting and bias information. The format is as follows:

```
DAMP    ATOL
1.0D-00 1.0D-03     least squares damping factor and solution (convergence) tolerance (3f8.0)
MAGWIN  FZWIN   TFWIN
1.0D0   1.0D0   1.0D0  the outlier exclusion limits for each data type (3f8.0)
MAGBIAS FZBIAS  TFBIAS
1.0D0   1.0D0   1.0D0  the relative biases for each data type (3f8.0)
```

The poles file must be called poles_01.dat and must be in the following format.

```
antafr 3
      8.00    -40.00      0.50   c2A
      8.00    -40.00      1.40    c5
      8.00    -40.00      3.40    c6
```

The first line contains the 6-character plate pair code, and the number of chrons the inversion is to be carried out for(a6,x,i2). The following lines contain the ‘test’ finite rotation poles for the inversion chrons - lat, lon, rotation angle, and the chron/anomaly identifier (3f10.0, 2x, a4).

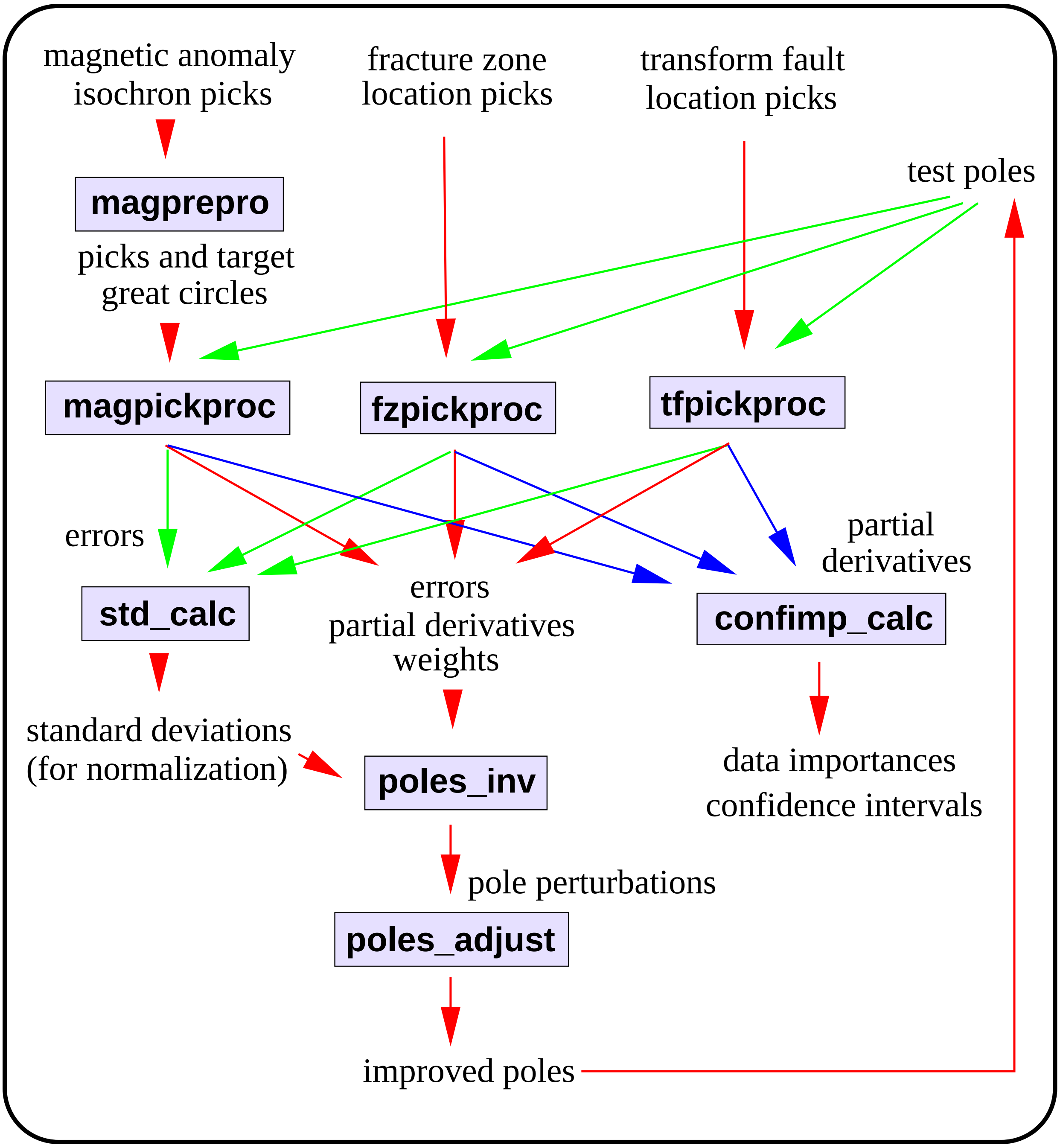


Figure 3.1: A flowchart showing the computational stages involved in the iterative inversion method.

The main input files must be labelled in accordance with the plate pair code given in the poles file. Thus, with the poles file above, indicating we are reconstructing the motion of the Antarctic and African plates, the magnetics input file is expected to be called **magpicks_antafr.dat**, the fracture zone list file called **fz_flist_antafr.dat**, and the transform fault pick file called **tfpicks_antafr.dat**.

3.2 Iterative procedure

A series of iterative inversions is to be carried out, until the solution has converged to within a certain tolerance. The iterative process is summarised by the flowchart shown in Figure 3.1, giving the names of the main programs used, and the main input/output of each routine. Further summaries of the programs can be found in Chapter 5.

The reconstruction methods have been designed to allow biasing of the inversion towards a particular type of data, but generally the only effect it appears to have is to make the inversion converge quicker with respect to the favoured set of data, without necessarily improving the overall fit. Thus the relative biases for each type of data are usually all set equal to 1 in **inv.dat**.

In practice it is best to set the data cutoff criteria in **inv.dat** to 5 ($\pm 5\sigma$) for the first five or ten iterations, so that virtually all of the input data are used. This value automatically causes all of the weighting systems described in Section 1.2 to be 'turned off' so that each datum is treated equally. This ensures that the choice of starting poles has minimal effect and the first few iterations lead the inversion towards the solution which satisfies the majority of the data. The cutoff criterion can then be reduced to a more useful lower value (and so returning weighting schemes to normal).

A typical command file to run a single iteration is shown below, and then the individual programs described in more detail in the next few sections. 111 and 222 represent the two plate ids, and ** the current iteration number.

```
#! /bin/sh
magpickproc
fzpickproc
tfpickproc
gstd_calc
confimp_calc3d
poles_inv
poles_adjust
sort_magerr
```

3.2.1 Data processing

magpickproc

Processes the magnetics data, calculating all of the errors and partial derivatives with respect to the current set of poles.

Input files:

magpicks_111222.dat
iter.dat
poles_**.dat

Output files:

rotmagpicks.dat: log file containing initial and final locations of rotated picks
mag_a.dat: partial derivative matrix
mag_b.dat: (-)misfits
mag_w.dat: weights
mag_loc.dat: pick locations and id's

fzpickproc

Processes the fracture zone data, calculating all of the errors and partial derivatives with respect to the current set of poles.

Input files:

fz_flist_111222.dat
set of fz files
iter.dat
poles_**.dat

Output files:

seeds.dat: log file containing final locations of seeds for each fz
flow_a.dat: partial derivative matrix
flow_b.dat: (-)misfits
flow_w.dat: weights
flow_loc.dat: pick locations and id's

tfpickproc

Processes the transform fault data, calculating all of the errors and partial derivatives with respect to the current set of poles.

Input files:

tfpicks_111222.dat
iter.dat
poles_**.dat

Output files:

tfplot.dat: log file containing closest points on fitted small circle for each transform section
tf_a.dat: partial derivative matrix
tf_b.dat: (-)misfits
tf_w.dat: weights
tf_loc.dat: pick locations and id's

3.2.2 Error analysis

gstd_calc

Calculates the mean and standard deviations of the distribution of errors for each datatype (assuming Gaussian distributions.

Input files:

mag_b.dat flow_b.dat tf_b.dat
inv.dat
iter.dat

Output files:

std.dat: mean and standard deviations for each dataset
stdconv.dat: file(appended to at each iteration) giving details of solution convergence

confimp_calc3d

Calculates the confidence ellipses(2-D) and ellipsoids(3-D) for the current solution poles, and the data importances for the input datasets.

Input files:

mag_a.dat flow_a.dat tf_a.dat
mag_b.dat flow_b.dat tf_b.dat
mag_w.dat flow_w.dat tf_w.dat
mag_loc.dat flow_loc.dat tf_loc.dat
inv.dat
std.dat
iter.dat
poles_**.dat

Output files:

confint_**.dat: 2-D confidence ellipses for each solution pole
confints_**.dat: 3-D confidence ellipsoids for each solution pole
mag_loci.dat flow_loci.dat tf_loci.dat: files containing pick (lat, lon, importance, id)

sort_magerr

Sorts all of the misfits so that all of the misfits created from a single magnetic pick (where it is rotated to several great circles) are summed together.

Input files:

mag_b.dat
mag_loc.dat

Output files:

mag_err.dat: summed misfits for each separate magnetic anomaly pick

3.2.3 Inversion

poles_inv

Combines the misfits and standard deviations from all three datasets in a single damped least-squares inversion

Input files:

mag_a.dat flow_a.dat tf_a.dat
mag_b.dat flow_b.dat tf_b.dat
mag_w.dat flow_w.dat tf_w.dat
mag_loc.dat flow_loc.dat tf_loc.dat
inv.dat
std.dat

Output files:

x.dat: set of solution perturbations to current poles
se.dat: standard errors from the least squares inversion
mag_wfin.datflow_wfin.dat tf_wfin.dat: final weight vectors

poles_adjust

Combines the misfits and standard deviations from all three datasets in a single damped least-squares inversion

Input files:

iter.dat
poles_**.dat
x.dat

Output files:

poles_(**+1).dat: next set of solution poles
iter.dat: incremented iteration number

4

3-plate reconstructions

The basic structure of the inversion procedure is the same as for the 2-plate inversion procedure, but with 3 times the number of input datasets. There are some changes to the actual inversion calculations and mechanics, principally that only 2 sets of independent pole parameters exist. This means that partial derivatives for the third plate pair must be determined with respect to the previous two sets of pole parameters. This is covered in more detail in Chapter 1.

4.1 Required input files

The iteration file **iter.dat** and the inversion data file **inv.dat** are the same as before. However, as we are dealing with a 3-plate system, the poles file (called 3poles_01.dat) now contains three sets of poles, for example:

```
samafr 3
   58.5343  -39.8922    0.7670   c2A
   59.9360  -39.6304    3.1180    c5
   58.0012  -37.0517    7.0450    c6
afrant 3
   -6.6868  140.8152    0.3889   c2A
   -6.7931  139.0326    1.4756    c5
  -10.5188  139.9158    3.2416    c6
samant 3
   88.6475  -50.8057    0.6091   c2A
   87.3107   -5.6210    2.5260    c5
   83.1638   -7.7901    5.4128    c6
```

The formatting is the same as for the 2-plate poles file. Although the third set of poles is included (in this case sam-ant), it is not actually used by the inversion procedure.

As with the 2-plate method, the input data files need to be labelled according to the plate pair codes included in the poles file, therefore for the above poles file, the following data files would be expected to exist: **magpicks_samafr.dat**; **fz_flist_samafr.dat**; **tfpicks_samafr.dat**; **magpicks_afrant.dat**; **fz_flist_afrant.dat**; **tfpicks_afrant.dat**; **magpicks_samant.dat**;

fz_flist_samant.dat; **tfpicks_samant.dat**.

4.2 Iterative procedure

Some of the programs used for the 3-plate reconstructions are the same as those for the 2-plate programs, and others have many subroutines in common - in the data processing programs (*i.e.* **fzpickproc**, the datasets for the first two plate pairs are processed exactly as before. A standard command file for a single iteration is shown below.

```
#! /bin/sh
3p_magpickproc
3p_fzpickproc
3p_tfpickproc
gstd_calc
3p_confimp_calc3d
poles_inv
3poles_adjust
sort_magerr
```

All of the files have the same input and output files as before except for the following exceptions (111 222 and 333 represent the three plate ids, and ** the current iteration number).

3p_magpickproc Now takes input pick data for 3 plate pairs instead of just the one: magpicks_111222.dat ; magpicks_222333.dat ; and magpicks_111333.dat .

3p_fzpickproc Now takes input pick data for 3 plate pairs instead of just the one: fz_flist_111222.dat ; fz_flist_222333.dat ; and fz_flist_111333.dat .

3p_tfpickproc Now takes input pick data for 3 plate pairs instead of just the one: tfpicks_111222.dat ; tfpicks_222333.dat ; and tfpicks_111333.dat .

3p_confimp_calc3d Reads in the 3-plate poles file 3poles_**.dat . Outputs confidence intervals for the first two plate pairs only: confint_111222_**.dat ; confints_111222_**.dat ; confint_222333_**.dat ; and confints_222333_**.dat .

3poles_adjust Adjusts the 3-plate poles file 3poles_**.dat to output 3poles_(**+1).dat.

5

Summaries of reconstruction programs and subroutines

magprepro.f : Inputs a set of magnetic anomaly isochron picks and according to the users choice, fits great circles to anomaly groups, and for each pick within a ridge section, decides which great circle(s) the pick will be rotated to during th reconstruction procedure. Outputs a processed isochron data file, ready for use with **magpickproc**.

magpickproc.f : Inputs a set of preprocessed magnetic anomaly isochron picks and great circles, and a set of reconstruction poles. The misfits of the data for the given poles, and the partial derivatives with respect to the poles are calculated and outputted as data files.

fzpickproc.f : Inputs a set of picks of fracture zone locations, and a set of reconstruction poles. The misfits of the data for the given poles, and the partial derivatives with respect to the poles are calculated and outputted as data files.

tfpickproc.f : Inputs a set of picks of transform fault locations, and a set of reconstruction poles. The misfits of the data for the given poles, and the partial derivatives with respect to the poles are calculated and outputted as data files.

std_calc.f : Calculates the mean and standard deviation for each of the reconstruction dataset error files, and outputs to a file *std.dat* used in later stages of the inversion, and a file *stdconv.dat* which stores a list of the means and standard deviations for each iteration, along with a chi-square statistic, giving an indication of the convergence of the inversion.

gstd_calc.f : Similar to **std_calc**, but on the assumption that the errors have normal distributions, determines the best-fit gaussian statistics for each set of errors. Outputs the same files as **std_calc**.

poles_inv.f : Takes the error and partial derivative files for each dataset and, after normalisation and weighting, runs a damped least-squres inversion, outputting a set of peturbations to the current set of poles.

confimp_calc.f : From the files of partial derivatives of each dataset, determines the confidence ellipses for the currrent set of poles and outputs these into a file. Also calculates the data importances for each dataset.

confimp_calc3d.f : Similar to **confimp_calc.f**, but calculates 3-d confidence ellipsoids for the current set of poles.

poles_adjust.f : Takes a set of pole peturbations produced by **poles_inv** and adjusts the current set of poles accordingly.

sort_magerr.f : Takes the file containing the full set of errors for the magnetics dataset, and collates the errors for the picks which have been used to create several datum. A file is outputted which has only a single RMS error for each isochron pick used.

magdat_count.f : Given a data file of isochron picks for a reconstruction, determines the number of picks fror each isochron.

magcheck.f : Given a set of rotation poles and a reconstruction dataset, rotates each magnetic anomaly pick back to the notional ridge axis at which it was formed.

calcrate.f : Given the location of a point on a ridge axis and a current rotation pole, calculates the local spreading rate and azimuth.

calcrates.f : Given the location of a point on a ridge axis and a set of rotation poles, calculates a series of spreading rates and azimuths for the spreading history covered by the given rotation poles.

3p_magpickproc.f : The 3-plate version of **magpickproc**, taking a set of reconstuction poles for a 3-plate system, and calculating the misfits and partial derivatives of the magnetic anomaly isochrons for each plate pair.

3p_fzpickproc.f : The 3-plate version of **fzpickproc**, taking a set of reconstuction poles for a 3-plate system, and calculating the misfits and partial derivatives of the fracture zone locations for each plate pair.

3p_tfpickproc.f : The 3-plate version of **tfpickproc**, taking a set of reconstuction poles for a 3-plate system, and calculating the misfits and partial derivatives of the transform fault locations for each plate pair.

3poles_inv.f : The 3-plate version of **poles_inv**.

3pconfimp_calc2.f : The 3-plate version of **confimp_calc3d**.

3poles_adjust.f : The 3-plate version of **poles_adjust**.

5.1 Reconstruction subroutines

5.1.1 Main subroutines

sub_openfile.f : Reads a filename from keyboard entry and opens the file as a logical unit, according to the requirements (read/write/new/old *etc.*).

sub_readpolesfile.f : Reads a file containing a set of finite rotation poles for a plate pair, and stores them in an array.

sub_read3polesfile.f : Reads a file containing the finite rotation poles for a 3-plate set. Stores first two sets of poles in arrays, but ignores third, dependent set.

sub_calc3rdpoles.f : From 2 sets of reconstruction poles for A↦B and B↦C, calculates the third, dependent set of reconstruction poles A↦C.

sub_magpickproc.f : The basis of programs **magpickproc** and **3p_magpickproc**, determines the errors and partial derivatives for a complete set of magnetic anomaly isochrons for one plate pair.

sub_gcfit.f : Determines the best-fit great circle passing through a set of locations on a sphere.

sub_dcmake.f : Builds a correlation matrix for a series of points which is used to determine the best-fit great circle.

sub_sortpnts.f : Given a series of locations on a sphere, sorts them into a progressive sequence along an axis joining the two-furthest separated points. Used for arranging in order the closest points on a great circle for the set of points used to generate the best-fit great circle.

sub_magpickcalc.f : Given a group of picks for a certain isochron, and the conjugate great circle to which to rotate them, calculates the error and partial derivatives for each datum. Uses **epdcalc**.

sub_magpickcalc_2pol.f : Given a group of picks for a certain isochron, and a non-conjugate target great circle to which to rotate them, calculates the error and partial derivatives for each datum. Uses **epdcalc**.

sub_magpickcalc3p.f : The 3-plate version of **sub_magpickcalc** used when processing data for the third plate boundary A↦C. The partial derivatives are calculated with respect to the 2 independent sets of poles, A↦B, and B↦C.

sub_magpickcalc3p_2pol.f : The 3-plate version of **sub_magpickcalc_2pol** used when processing data for the third plate boundary A↦C. The partial derivatives are calculated with respect to the 2 independent sets of poles, A↦B, and B↦C.

sub_epdcalc.f : Given a single rotated isochron location and a target great circle, calculates the misfit, and the partial derivative of the error with respect to the position of the rotated point.

sub_fzpickproc.f : The basis of programs **fzpickproc** and **3p_fzpickproc**, determines the errors and partial derivatives for a complete set of fracture zone locations for one plate pair.

sub_readpickfile.f : Reads in a file containing a set of picks for a fracture zone, along with the relevant seed information.

sub_fpoles2stage.f : From a set of finite reconstruction poles, calculates the 2 sets of stage poles, for each flank of the plate pair.

sub_flowline.f : From a set of reconstruction poles, determines a flowline from a given point.

sub_seedinv.f : Given a set of fracture zone locations and estimated seed point, adjusts the seed point (within limits) to provide a best-fit flowline for the set of picks.

sub_fzpickcalc_seed.f : Calculates the misfits of fracture zone locations from a test flowline, and the partial derivatives with respect to the seed point of the flowline. Used in the determination of the seed point which gives the best-fit flowline for a particular fracture zone.

sub_dqr.f : Determines the solution to an over-determined least-squares problem $\mathbf{Ax} = \mathbf{b}$. Used in the adjustment of seed points for flowlines in **sub_seedinv**.

sub_pdflowseedx.f : Determines the changes in a flowline with respect to changes in the zero-age seed.

sub_fzpickcalc.f : For a set of picks along a fracture zone, determines the misfits from a given flowline, and the partial derivatives with respect to the finite rotation poles used to generate the flowlines.

sub_fzpickcalc3p.f : The 3-plate version of **sub_fzpickcalc** used when processing data for the third plate boundary A$\mapsto$C. The partial derivatives are calculated with respect to the 2 independent sets of poles, A$\mapsto$B, and B$\mapsto$C.

sub_tfpickproc.f : The basis of programs **tfpickproc** and **3p_tfpickproc**, determines the errors and partial derivatives for a complete set of transform fault locations for one plate pair.

sub_std.f : For a series of data values, calculates the mean and standard deviation.

sub_gauss.f : Calculates the best-fit Gaussian distribution for a set of data.

sub_writepolesfile.f : Writes a set of finite rotation poles for a plate pair to file.

sub_write3polesfile.f : Writes the finite rotation poles for a 3-plate set to file.

sub_dfgcalc.f : From the covariance matrices for the reconstruction poles A$\mapsto$B and B$\mapsto$C, determines the matrices D, F and G required to calculate the covariance matrix for the resultant A$\mapsto$C reconstruction poles (given no data input from A$\mapsto$C plate boundary).

sub_mtxmul.f : Carries out the basic matrix multiplication $\mathbf{C} = \mathbf{BA}$.

sub_mtxvtr.f : Carries out the basic matrix operation $\mathbf{y} = \mathbf{Ax}$.

sub_writematrix.f : Writes a 2-dimensional array to file.

sub_writevector.f : Writes a 1-dimensional array to file.

sub_int2char.f : Simple routine converting an integer between zero and ninety nine into character*2 format (used in creating the incremental filenames for successive iterative inversions).

sub_intch.f : Simple routine converting an integer between zero and ninety nine into a character variable of the same length.

sub_strcat.f : Concatenates two character arrays, skipping any blank spaces. Used of creating filenames when supplied with a directory name and file name separately.

5.1.2 Geometric operations and partial derivative subroutines

sub_pnt2vec.f : Converts a point location in longitude and latitude to a 3-vector on a unit sphere.

sub_normvec.f : Normalises a vector to represent a vector on a unit sphere.

sub_vec2pnt.f : Converts a 3-vector on a unit sphere to a point location in longitude and latitude.

fun_vecdprod.f : Calculates the dot product of two vectors.

sub_vecxprod.f : Calculates the vector (cross) product of two vectors.

fun_pangsep.f : Calculates the angular seperation of two points on a sphere.

fun_vangsep.f : Calculates the angular seperation of two vectors.

fun_angpdcalc.f : Calculates the change in angular seperation of two vectors when a peturbation is applied to one of the vectors.

sub_pol2mat.f : Converts a rotation pole in the form of latitude, longitude and angle into a 3x3 rotation matrix.

sub_mat2pol.f : Determines the pole parameters, latitude, longitude and angle from the corresponding 3x3 rotation matrix.

sub_sumrots.f : Combines two rotation matrices $\mathbf{A}$ and $\mathbf{B}$ to obtain a third, $\mathbf{C} = \mathbf{BA}$.

sub_addmats.f : Adds two matrices $\mathbf{A}$ and $\mathbf{B}$ to obtain a third, $\mathbf{C} = \mathbf{A} + \mathbf{B}$.

sub_add3mats.f : Adds three matrices $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$, to obtain a fourth, $\mathbf{D} = \mathbf{A} + \mathbf{B} + \mathbf{C}$.

sub_matscal.f : Scales a matrix $\mathbf{A}$ by a constant c, such that $\mathbf{A}' = c\mathbf{A}$.

sub_matxvec.f : Carries out the basic matrix operation $\mathbf{y} = \mathbf{Ax}$ for the 3x3 matrix $\mathbf{A}$ and 3-vectors $\mathbf{x}$ and $\mathbf{y}$.

sub_rotatepnt.f : Given a rotation matrix $\mathbf{R}$ and a vector $\mathbf{p}$, calculates the rotated vector (on a unit sphere) $\mathbf{p}'$.

sub_rotatevec.f : Given a rotation matrix $\mathbf{R}$ and a point location on a sphere (ϕ, θ), calculates the rotated point on the sphere, (ϕ', θ').

sub_diamg.f : Given a point location on a sphere, calculates the antipode of this point on the sphere.

sub_pnt2pdvecs.f : Given a point location on a sphere, calculates the partial derivatives of the 3-vector representation of the point with respect to latitude and longitude.

sub_pol2pdmats.f : Given the pole parameters of a rotation, (latitude, longitude, angle), determines the partial derivatives of the rotation matrix with respect to each of these parameters.

sub_mats2polpds.f : Given a rotation matrix, and the partial derivative of this matrix with respect to an unknown peturbation, calculates the partial derivatives of the correpsonding pole parameters with respect to the unknown peturbation.

sub_mats2vecpds.f : Given a rotation matrix, and the partial derivative of this matrix with respect to an unknown peturbation, calculates the partial derivatives of the 3-vector repsentation of the corresonding rotation pole with respect to the unknown peturbation.

sub_poles2stpds.f : From a set of finite rotation poles, calculates the partial derivatives of each stage pole (in vector form) on both flanks, with respect to the finite rotation pole parameters.

sub_x2poles2stpds.f : The 3-plate version of **sub_poles2stpds**, where the stage poles for the third plate pair are dependant on the finite rotation poles of the two previous plate pairs.

sub_stpdarraybld.f : Used in **sub_poles2stpds** and **sub_x2poles2stpds** to build the arrays of partial derivatives for the stage poles. Calculates the partial derivative for one stage pole and stores it in the specified location within an array.

sub_rotnpd2scpd.f : A complex routine which calculates the partial derivative of a rotation matrix $\mathbf{C}$ with respect to an unknown pole parameter, where $\mathbf{C}$ represents a rotation around the same pole as for the rotation matrix $\mathbf{S}$ (which is the sum of the two rotation matrices, $\mathbf{A}$ and $\mathbf{B}$) but with a scaled rotation angle. The effect of the peturbation is supplied as a peturbation to one of the rotation matrices $\mathbf{A}$ or $\mathbf{B}$.

5.2 Programs for data manipulation and display

addrot.c : A simple test program which reads two rotation poles from the keyboard, and displays the resultant on screen.

stagepoles.c : Reads a set of finite rotation poles from file and then calculates the set of stage poles for each flank of the plate pair. The stage poles are displayed on screen and outputted to a file in a format suitable for the **gmt** routines **psxy** and **psvelomeca**.

tripcalc.c : Given a set of finite rotation poles for a 3-plate system, the current location of the triple junction between the three plates, and a tolerance value for the RFF configuration, calculates a series of velocity triangles for several times in the past. The likely configuration and motion of the triple junction over this time period is calculated.

controt.c : Given a set of finite rotation poles for a 3-plate system, and a continental outline/coastline file, this program rotates the relevant continents to their positions at each poles chron, whilst keeping one of the plates fixed.

calcell.c : Reads in a file containing information about the confidence ellipses for a set of rotation poles, and calculates a loci of points on the surface of the earth, representing the limits of a specified confidence interval.

calcellsurf.c : Similar to **calcell**, but reads in information about the 3d confidence ellipsoids for a set of rotation poles and calculates the upper surface of the ellipsoids intersection with the surface of the earth, representing the limits of a specified confidence interval.

create_data.c : Generates 1-d profiles for (a) Geoid steps across a fracture zone or (b) simple topography models - either Gaussian or straight lines.

multi_fzgeoid.c : Generates a 1-d geoid profile across a series of up to 4 fracture zones with specified ages and offsets.

data2grav.c : Converts 1-d topographic or geoid profiles into gravity anomalies.

create_gauss.c : Given the parameters of a Gaussian distribution, generates a series of points which can be used to plot the distribution.

flowlines.c : Reads in a set of finite rotation poles form a file and displays on screen the corresponding sets of stage poles for each flank.

makeprofs.c : Reads a set of approximate locations along a fracture zone from a database file, and uses these to generate a series of perpendicular profiles across the fracture zone. Gravity values along the profiles are then sampled from a specified gravity grid, and if required, the minimum picking routine **pick_min** is run. This can be more generally used for generating perpendicular profiles across any feature, and sample from any kind of grid (gravity, topography, magnetics *etc.*).

pick_min.c : Given a series of perpendicular profiles across a fracture zone, this routine attempts to find a minimum for each profile, representing the centre of the fracture zone valley. The output is in the form of a file suitable for use in the program **xfzpick** and a log file recording information for each profile.

mgd77toxyz.c : Reads in a data file in mgd77 format, and outputs the required type of data (magnetics, gravity or topography) in ASCII x,y,value format, within specified limits or regions if desired. If the mgd77 file contains data from more than one cruise, then the data can be written out as a single file, separate files for each cruise, and/or only from specific cruises.

manu_data.c : Given a set of finite rotation poles and a series of ridge-transform intersections, generates a set of synthetic fracture zone locations and magnetic anomaly isochrons, with intrinsic random errors if required. Outputted in file formats suitable for use in reconstruction routines, in order to be used for testing reconstruction methods.

ridgeflow.c : Given a set of finite rotation poles and a series of points representing the location of the present ridge axis, generates a series of rotated locations of the ridge axis for each rotation chron.

mpicks2recin.c : Takes a set of magnetic anomaly isochron picks made from the program **xmagpick** and converts them into the format required for the reconstruction routines.

mpicks2screen.c : Takes set(s) of magnetic anomaly isochron picks made from the program **xmagpick** and generates a command file to display them as a postscript file. The different anomalies are represented according a specified file containing a symbol table. Options are included to limit the display to a certain region, to plot the current ridge axis and/or fracture zones, and also to plot the ship tracks from which the picks were made.

rpicksort.c : Given a set of magnetic anomaly files in reconstruction format and a geographical region, outputs a list if picks which are contained within the area. Used for sorting the picks before an inversion, but since superseded by **xsortdata**.

rpix2screen.c : Reads 1 or more files containing magnetic anomaly isochron picks in reconstruction format and generates a command file to display them as a postscript file. The different anomalies are represented according a specified file containing a symbol table. Options are included to limit the display to a certain region and to plot the current ridge axis and/or fracture zones.

split_fzpicks.c : Take an output file from **xfzpick** and writes the fracture zone picks either into a single file, or different types into separate files, splitting each into multiple segments if a specified inter-pick distance is exceeded. The format of the output file is either in (x,y) for use with **psxy**, or in a format for use in the reconstruction routines.

split_magpicks.c : Take an output file from **xmagpick** and writes the magnetic anomaly isochron picks into separate files for each different isochron.

rdata_replace.c : Reads in a magnetics reconstruction dataset, and a magnetics file from a single cruise. If any picks within the cruise file have been changed, added, or removed, then this program updates the reconstruction dataset. This avoids having to rebuild the reconstruction file from scratch, from the cruise files, if any picks are adjusted, so leaving grouping and organisation of the picks within ridge sections unchanged.

ipix2screen.c : Reads the magnetic isochron data output file from the inversion methods, and generates a command file to display the original picks, target great circles, and rotated picks as a postscript file. The original and/or rotated anomaly picks are represented according to different specified files containing symbol tables. Various selection and display options are included, such as drawing the data errors and/or importances on the same plot.

quantile.c : This program reads in a series of (random) data sample values and calculates points for a quantile-quantile plot of the dataset, using a Gaussian test function with a mean of 0 and standard deviation 1.

xycrop.c : Reads a series of $x, y(, z \dots)$ data points from a data file, or **stdin** and only outputs the points which lie within the desired range.

xysplit.c : Reads a series of $x, y(, z \dots)$ data points from a data file, or **stdin**, and splits it into continuous ascending/descending sequences with respect to x, (with a specified tolerance to account for possible random errors), splitting the sequences further if a specified data gap is exceeded, and only outputting data above a specified minimum segment length.

xyfilter.c : Reads in a continuous ascending/descending sequence of (x, y) data points, and filters the data in the time domain. The data can be resampled onto a regular spacing if required, and either a mean offset or a trend is removed before the profile is tapered and filtered. The filter can be hi-pass, lo-pass, or both.

add_data.c : Takes two (x, y) profiles and adds the data points for any common values, and outputting these data points.

rms_data.c : Reads in two profiles of (x, y, z) data points and calculates the RMS difference between the profiles.

samplecalc.c : Calculates the minimum, maximum, mean and standard deviation for a series of sample values.

xyfit.c : Reads a series of x, y data points from a data file, or **stdin** and uses linear regression to determine a best fit line, and outputs points for plotting said line.

5.3 UNIX/AWK scripts

getpicks : Given a file containing a list of fracture zone files, as used in the reconstructions, reads in these files and outputs the full set of data in a single x-y file suitable for plotting.

getseeds : Given a file containing a list of fracture zone files, as used in the reconstructions, reads in these files and outputs the reference seed point for each fracture zone into a single x-y file suitable for plotting.

diffpoles/diff3poles : Given two files containing sets of finite rotation poles, determines the differences between the two.

rmsdiffpoles/rmsdiff3poles : Given two files containing sets of finite rotation poles, determines the RMS differences for latitude, longitude and rotation angle between the two.

rmsdiff_allpoles/rmsdiff_all3poles : Calculates the RMS difference between each successive pair of finite rotation poles for an iterative 2/3 plate inversion.

tripvel2plot : From an output file from **tripcalc** creates a script to plot the series of velocity triangles in a single postscript file.