Strain

Jonathan M. Lees
University of North Carolina, Chapel Hill
Department of Geological Sciences
CB #3315, Mitchell Hall
Chapel Hill, NC 27599-3315
email: jonathan.lees@unc.edu
ph: (919) 962-0695

November, 2010

1 Playing with Strain

2 Introduction

The following are a set of notes about continuum mechanics that show students how to "play" with strain for deeper understanding of mathematics and manipulation of matrices and tensors.

The text here is not intended to replace a text from a standard book on the subject but rather to augment it. We expect that students are already learning the theoretical and applied subjects from books and class room lectures. Here we show how to apply some of the concepts and illustrate with graphical implementation.

Using a computer program to play with stress and strain gives a student a way to get a feel and intuition for the relationships of the various parameters used in describing these physical components. If the student can change a parameter and see how it effects the outcome, they can be better equipped to seeing patterns in field outcrops and hand specimens. Insight can be gained by 'fiddling', so we encourage users to modify and perturb, to experiment with the input and output to see what happens when things change.

Laboratory experiments on the effects of changing stress and strain can be expensive and time consuming. Computerized experiments, on the other hand, can reveal relationships in a *gedanken* (thought experiment) way than may provide the necessary *experience* students need to assess geological processes.

3 The Yellow Notepad

Pure shear, simple shear and pure strain are all represented mathematicall as matrix transformations. In two dimensions this is a 2 by 2 matrix that operates on a point or a set of points that represent a body in 2D space. The difference between the three different regimes is seen in the difference in the structure of their respective matrix transformations. We represent these architypal deformations in cartoon form in Figure 1:

In three dimensions deformation is represented by a 3 by 3 matrix, of course - but it is more difficult to graphically illustrate three dimensional deformation, so for now we consider simple 2 deformation.

3.1 Pure Shear

Pure shear is the result of a body reacting to a normal stress applied along the y-axis. The body deforms in the y and x-directions. The formula for transformation is:

$$\boldsymbol{n}' = \boldsymbol{n}\boldsymbol{T} = [n_1 n_2] \begin{bmatrix} 1+\epsilon_1 & 0 \\ 0 & \frac{1}{1+\epsilon_1} \end{bmatrix} \tag{3.1}$$

As an example in R consider the following. We set the variable ϵ, create the matrix and perform the transformation on a set of points (pts) that form a unit, square block. The Figure 1A shows the block in the undeformed state prior to deformation and in the deformed state post application of the deformation matrix, $\boldsymbol{T}$. Note that the origin of the block is at point (0,0) which is not deformed by the transformation because translation is not included in the matrix.

```
> pts = cbind(c(0, 0, 1, 1), c(0, 1, 1, 0))
> epsilon1 = 0.2
> H = matrix(c(1 + epsilon1, 0, 0, 1/(1 + epsilon1)), ncol = 2)
> NP = pts %*% H
```

Which is illustrated in Figure 1A. The plotting instructions used for this part of the figure are in the following code chunk:

```
> rx = range(c(pts[, 1], NP[, 1]))
> ry = range(c(pts[, 2], NP[, 2]))
> plot(rx, ry, type = "n", asp = 1, ann = FALSE, axes = FALSE)
> lines(c(pts[, 1], pts[1, 1]), c(pts[, 2], pts[1, 2]))
> lines(c(NP[, 1], NP[1, 1]), c(NP[, 2], NP[1, 2]), col = "red")
> title("A. Pure Shear")
```

3.2 Pure strain

Pure strain is like a stretching of the box. In this case there are two parameters which describe the deformation ϵ_1 and ϵ_2. Each of these represent deformation, or stretching, in different directions, x and y.

$$\boldsymbol{n}' = \boldsymbol{nT} = [n_1 n_2] \begin{bmatrix} 1+\epsilon_1 & 0 \\ 0 & 1+\epsilon_2 \end{bmatrix} \tag{3.2}$$

As an example in R consider the following:

```
> pts = cbind(c(0, 0, 1, 1), c(0, 1, 1, 0))
> epsilon1 = 0.2
> epsilon2 = 0.3
> H = matrix(c(1 + epsilon1, 0, 0, 1 + epsilon2), ncol = 2)
> NP = pts %*% H
```

Which is illustrated in Figure 1B.

3.3 Simple Shear

Simple shear occurs when a shearing force is applied across the (top) side of the block along the x-direction.

$$\boldsymbol{n}' = \boldsymbol{n}\boldsymbol{T} = [n_1 n_2] \begin{bmatrix} 1 & \tan\phi \\ 0 & 1 \end{bmatrix} \tag{3.3}$$

As an example in R consider the following:

```
> pts = cbind(c(0, 0, 1, 1), c(0, 1, 1, 0))
> phi = 30 * pi/180
> H = matrix(c(1, tan(phi), 0, 1), ncol = 2)
> NP = pts %*% H
```

Which is illustrated in Figure 1C.

Another example illustrating simple shear in a classic application showing a synthetic Brachiopod can is presented in the following.

Consider a hand sample that contains the fossil remains of six Brachiopods. They are arranged here in arbitrarily in columns and rows for presentation purposes.

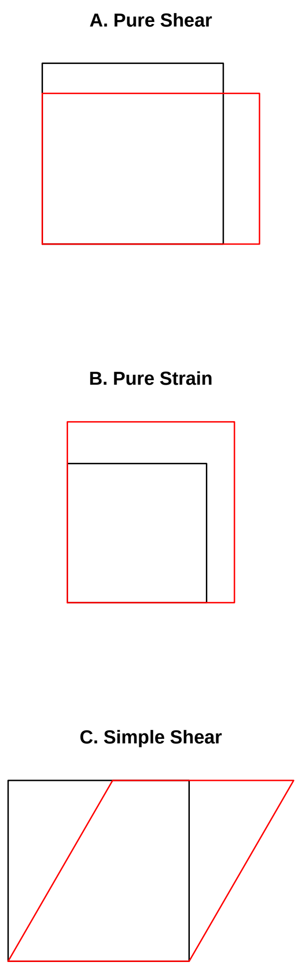

Figure 1: Simple shear, stress applied parallel to the x-axis

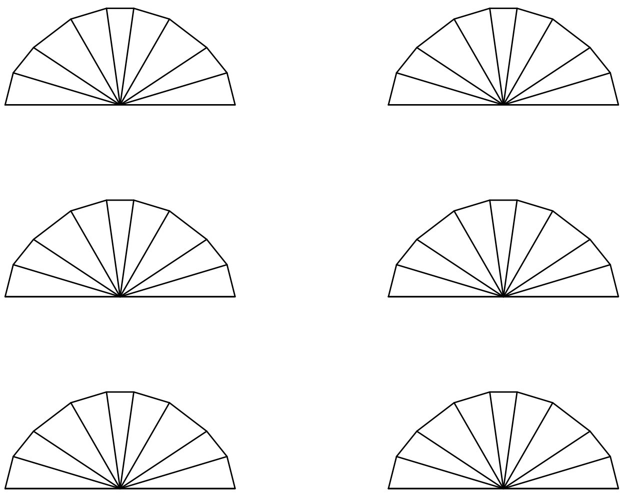

Figure 2: Simple shear strain applied parallel to the x-axis

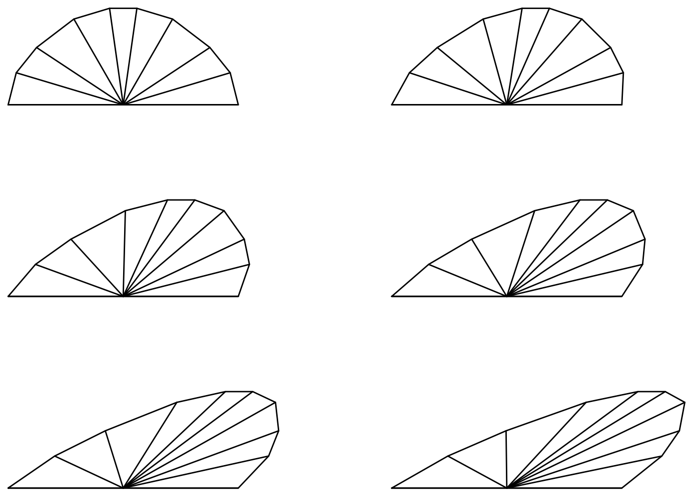

Figure 3: Simple shear strain applied parallel to the x-axis

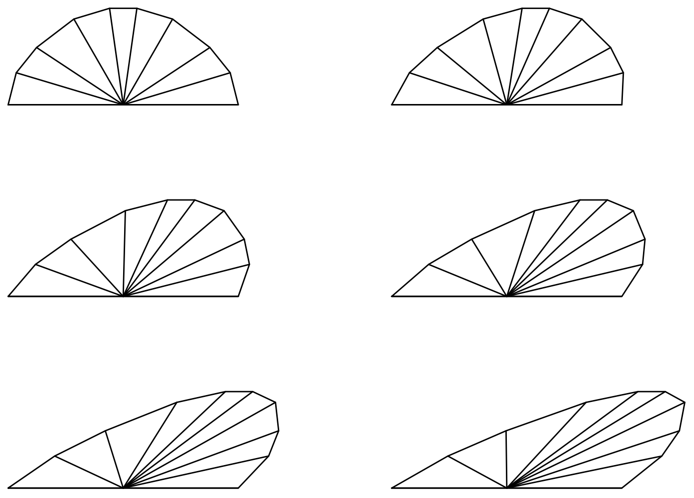

Figure 4: Simple shear strain applied parallel to the x-axis

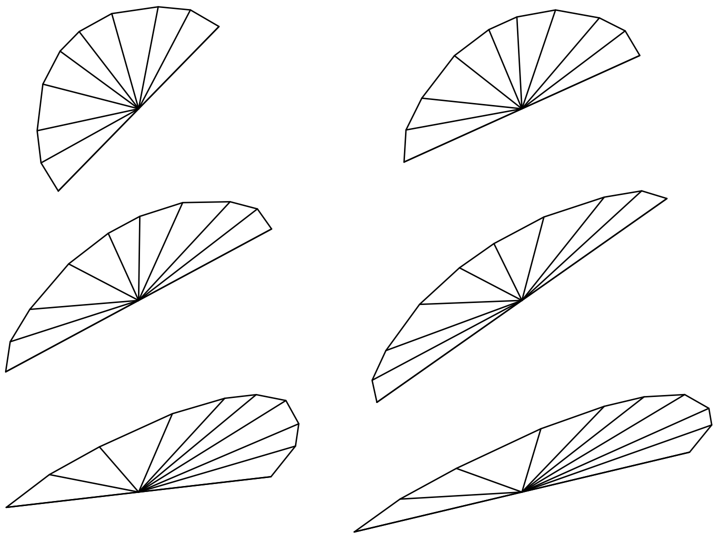

Figure 5: Simple shear strain applied parallel to the x-axis

4 Details and Advanced Graphics

4.1 Pure Shear

Pure shear arises when a body is subjected to a normal stress and is allowed to deform in a direction perpendicular to the normal stress. This is illustrated in the graphic shown in Figure 6 where the normal stress is shown as arrows pointing down along the z-axis and deformation is along the x-axis.

```
> e = 0.5
> pureshear = diag(c(1 + e, 1/(1 + e)))
> EX = seq(from = 0, to = 10, length = 15)
> mm = meshgrid(EX, EX)
> NEWpure = cbind(as.vector(mm$x), as.vector(mm$y)) %*% pureshear
> NEWmm = list(x = matrix(NEWpure[, 1], ncol = ncol(mm$x), nrow = nrow(mm$x)),
+     y = matrix(NEWpure[, 2], ncol = ncol(mm$y), nrow = nrow(mm$y)))
> rx = range(c(mm$x, NEWmm$x))
> ry = range(c(12, mm$y, NEWmm$y))
> plot(rx, ry, type = "n", asp = 1, xlab = "X-m", ylab = "Y-m",
+     axes = FALSE)
> u = par("usr")
> desh(mm, add = TRUE, PTS = FALSE, colmesh = grey(0.8))
> segments(mm$x, mm$y, NEWmm$x, NEWmm$y, col = rgb(1, 0.8, 0.8))
> desh(NEWmm, add = TRUE, PTS = FALSE, colmesh = rgb(0.2, 0.6,
+     0.2), lwd = 1.5)
> itop = u[4] - 0.1 * (u[4] - max(EX))
> arrX = EX[2:length(EX)] - diff(EX)/2
> fancyarrows(arrX, rep(itop, length(arrX)), arrX, rep(max(EX),
+     length(arrX)), thick = 0.08, headlength = 0.4, headthick = 0.2,
+     col = grey(0.5), border = "black")
> axis(1)
> axis(2, pos = -0.5)
> title("Pure Shear")
```

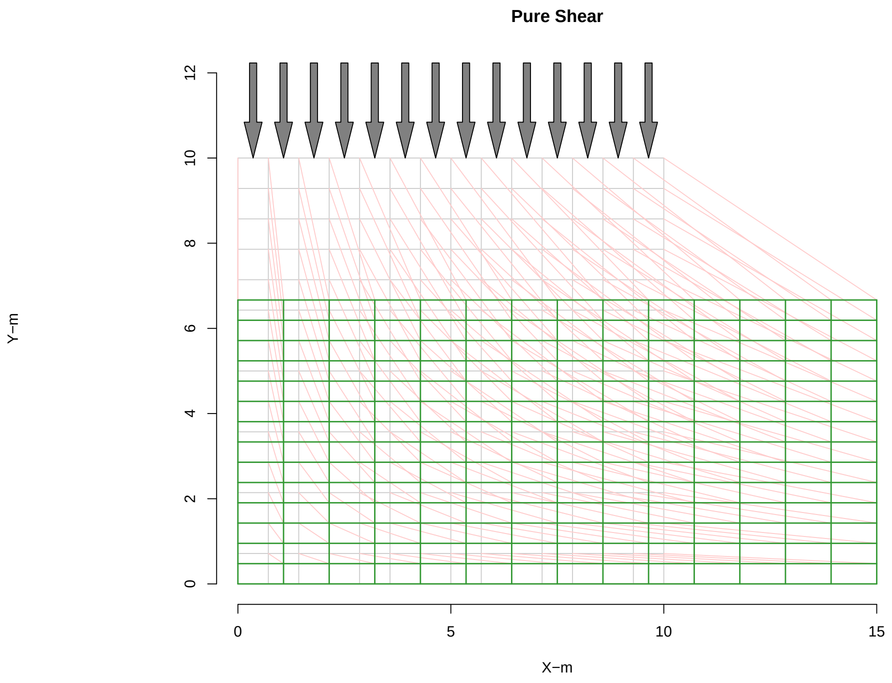

Figure 6: Pure shear represented normal stress in the z-axis and deformation in the x-axis. The grey is the original undeformed state, the dark green is the deformed state and the light pink show the displacement from undeformed to deformed state.

4.2 Simple Shear

Simple shear occurs when two blocks are pressed together and shear each other Figure 7.

To understand Simple shear we must examine the polar decomposition of a matrix.

4.3 Polar Decomposition

$$\boldsymbol{A} = \boldsymbol{U}\boldsymbol{P} \tag{4.1}$$

The Singular Value Decomposition (SVD) of A is

$$\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^t \tag{4.2}$$

and the Polar Decomposition of A is,

$$\begin{aligned} \boldsymbol{P} &= \boldsymbol{V}\boldsymbol{\Sigma}\boldsymbol{V}^t \\ \boldsymbol{U} &= \boldsymbol{U}\boldsymbol{V}^t \end{aligned}$$

The Polar decomposition can be retrieved from the matrix in R via the singular Value decomposition, or the svd.

```
> print(PolarDecomp)
```

```
function (A)
{
    E = svd(A)
    P = E$v %*% diag(E$d) %*% t(E$v)
    U = E$u %*% t(E$v)
    return(list(P = P, U = U))
}
```

Simple shear can be illustrated by multiple applications of the shear matrix:

```
> simpleshear = matrix(c(1, 0.2, 0, 1), ncol = 2)
> pts = 10 * cbind(c(0, 0, 1, 1), c(0, 1, 1, 0))
> LPTS = list()
> NP = pts
> MAX = list(x = 0, y = 0)
> for (i in 1:10) {
+     APTS = NP
+     NP = APTS %*% simpleshear
+     LPTS[[i]] = NP
+     MAX$x = max(c(MAX$x, NP[, 1]))
+     MAX$y = max(c(MAX$y, NP[, 2]))
+ }
```

```
> plot(c(0, MAX$x), c(0, MAX$y), type = "n", asp = 1, ann = FALSE,
+     axes = FALSE)
> u = par("usr")
> for (i in 1:length(LPTS)) {
+     NP = LPTS[[i]]
+     lines(c(pts[, 1], pts[1, 1]), c(pts[, 2], pts[1, 2]))
+     lines(c(NP[, 1], NP[1, 1]), c(NP[, 2], NP[1, 2]), col = "red")
+ }
> py = pts[2, 2] + 0.08 * (u[4] - pts[2, 2])
> fancyarrows(pts[1, 1], py, pts[3, 1], py, thick = 0.08, headlength = 0.4,
+     headthick = 0.2, col = grey(0.5), border = "black")
```

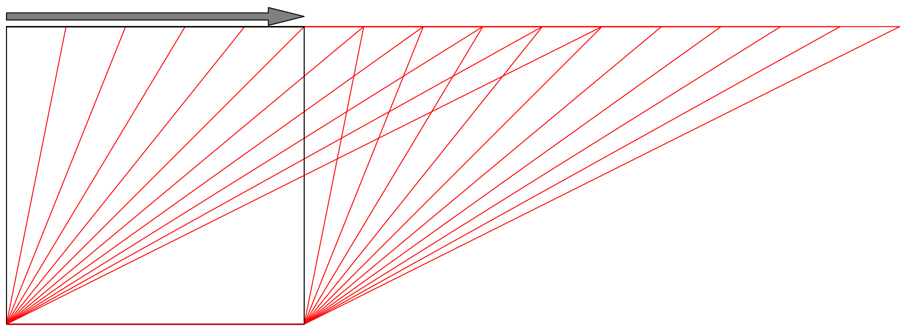

Figure 7: Simple shear, stress applied parallel to the x-axis

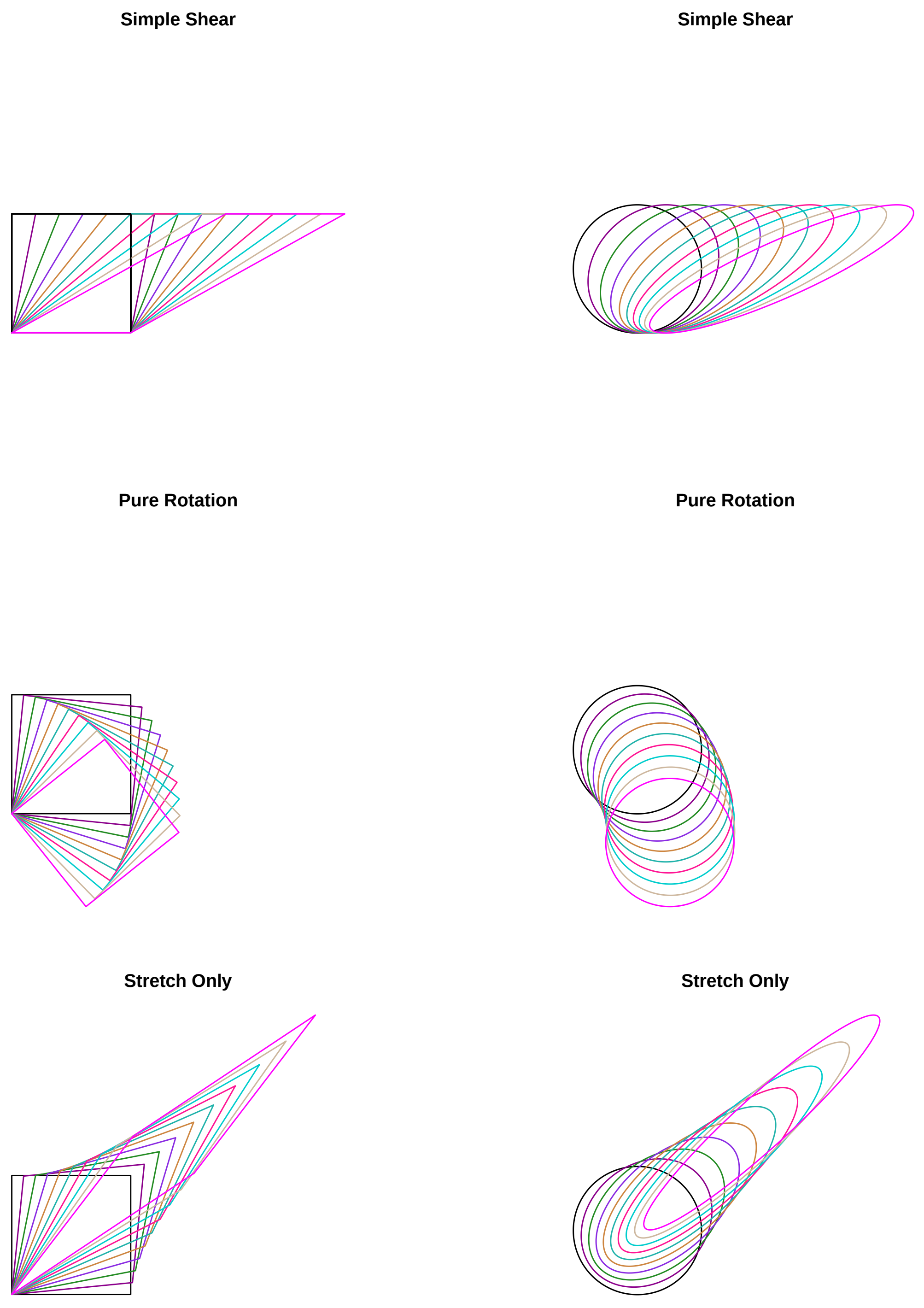

Figure 8: Break down of Simple shear into a Pure rotation and a stretch deformation. Two examples are shown with different starting shapes. The center of the cirlce is located at (5,5) so it is also shifted during deformation.

5 Inversion for Shear

Suppose we have a deformed biological organism that has been deformed from a previously known and specified shape. The deformation occurred post deposition, so we assume all deformation can be attributed to strains related to geologic processes.

We can estimate the total deformation, given the initial and final deformation by constructing an inverse problem.

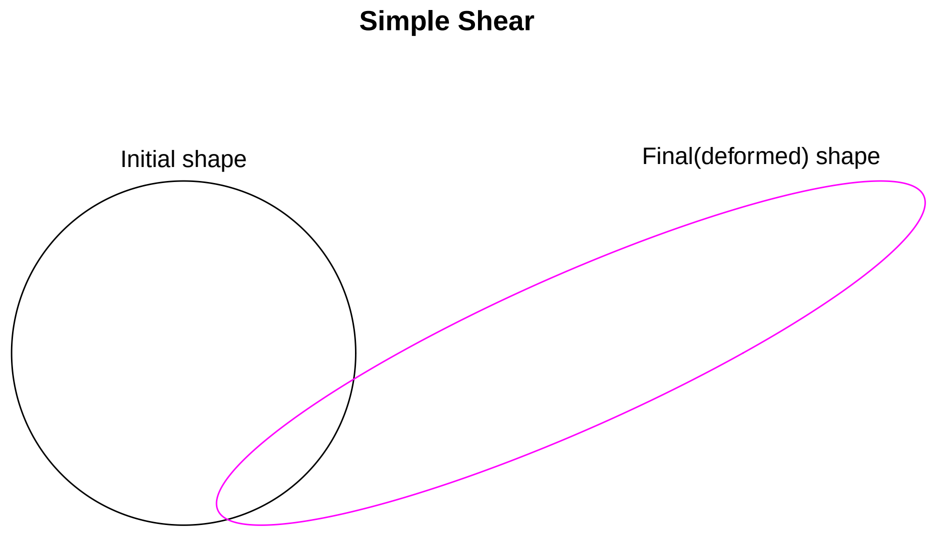

Figure 9: Initial and final shape of fossil oolith

As an example take the case of simple shear. we know that simple shear can be represented as a matrix multiplication,

$$\boldsymbol{n}' = \boldsymbol{n}\boldsymbol{T} = [n_1 n_2] \begin{bmatrix} 1 & \tan\phi \\ 0 & 1 \end{bmatrix} \tag{5.1}$$

However, if we turn this around we can find $\boldsymbol{T}$ given the two states $\boldsymbol{n}'$ and $\boldsymbol{n}$. To do this first we multiply by the transpose of $\boldsymbol{n}$, invert the square matrix and then multiply by the inverse.

$$\begin{aligned}
\boldsymbol{n}' &= \boldsymbol{n}\boldsymbol{T} \\
\boldsymbol{n}^t\boldsymbol{n}' &= \boldsymbol{n}^t\boldsymbol{n}\boldsymbol{T} \\
\left(\boldsymbol{n}^t\boldsymbol{n}\right)^{-1}\boldsymbol{n}^t\boldsymbol{n}' &= \left(\boldsymbol{n}^t\boldsymbol{n}\right)^{-1}\boldsymbol{n}^t\boldsymbol{n}\boldsymbol{T} \\
\left(\boldsymbol{n}^t\boldsymbol{n}\right)^{-1}\boldsymbol{n}^t\boldsymbol{n}' &= \boldsymbol{T} \qquad (5.2)
\end{aligned}$$

```
> T1 = t(NP1) %*% NP1
> DEF = solve(T1) %*% t(NP1) %*% NPF
```

We illustrate the process of inverting for strain by considering a synthetic example pertaining to estimating strain from crystolographic analysis.

To begin we create a handsample consisting of a random distribution of crystals magnified by a microscope. Each sample is then randomly rotated to simulate a real world situation, Figure 10.

Next the sample is deformed via one of the shear-strain methods described above. In this case we apply a simple shear to the rock, such that each sample is sheared with the same strain tensor. since each crystal is rotated to a different orientation, the resulting strained crystals appear to have a randome shearing applied.

The deformation matrix is defined in the usual way with a designated ϵ:

```
> epsilon1 = 0.4
> H = matrix(c(1 + epsilon1, 0, 0, 1/(1 + epsilon1)), ncol = 2)
> H = rbind(H, c(0, 0))
> H = cbind(H, c(0, 0, 1))
> print(H)
```

Given the hand sample on the right side of Figure 11, our task is to determine what tectonic shear occurred that deformed the host rock. We do not know, *a priori*, what the rotations were. In this case, we make the assumption that all the crystals were originally un-rotated - i.e. they formed or deposited, in the rock sample in the orientation depicted in Figure 10, left hand side.

Using this assumption, each crystal's location is noted and then translated to the origin. There an inversion is done to extract the associted polar decomposition.

First we invert each crystal:

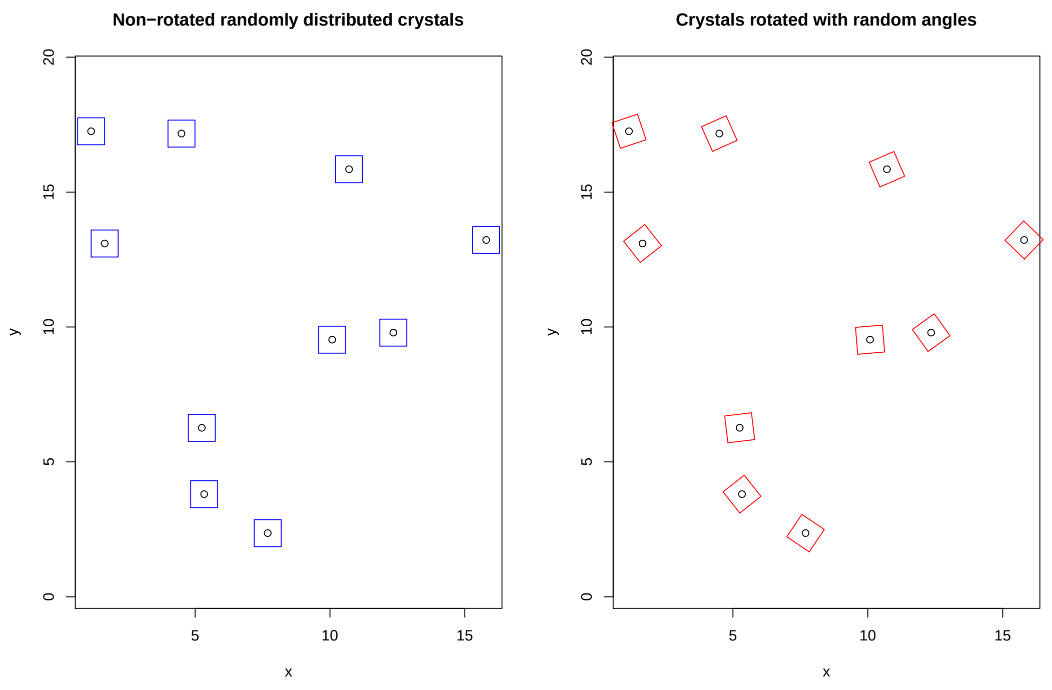

Figure 10: Example of a synthetic hand sample. On the left the handsample is shown as randomly distributed crystals, with no initial rotation. On the right, each crystal has a random rotation applied.

```
> NP = pts - 0.5
> DefList = list()
> for (i in 1:n) {
+     b = diag(3)
+     b[3, 1] = x[i]
+     b[3, 2] = y[i]
+     br = diag(3)
+     br[3, 1] = -x[i]
+     br[3, 2] = -y[i]
+     NPF = def2[[i]] %*% br
+     NPF = NPF[, 1:2]
+     T1 = t(NP) %*% NP
+     DEF = solve(T1) %*% t(NP) %*% NPF
+     DefList[[i]] = list(def = NA, P = NA, U = NA)
+     DefList[[i]]$def = DEF
+ }
```

The inversion matrices saved in the DefList list each represent the matrix $\boldsymbol{T}$ that will transform an $\boldsymbol{n}$ point into a deformed state $\boldsymbol{n}'$. Not knowing what the original orientation was, we assumed vertical orientations. The assumption of a common initial orientation in geology is not too unusual.

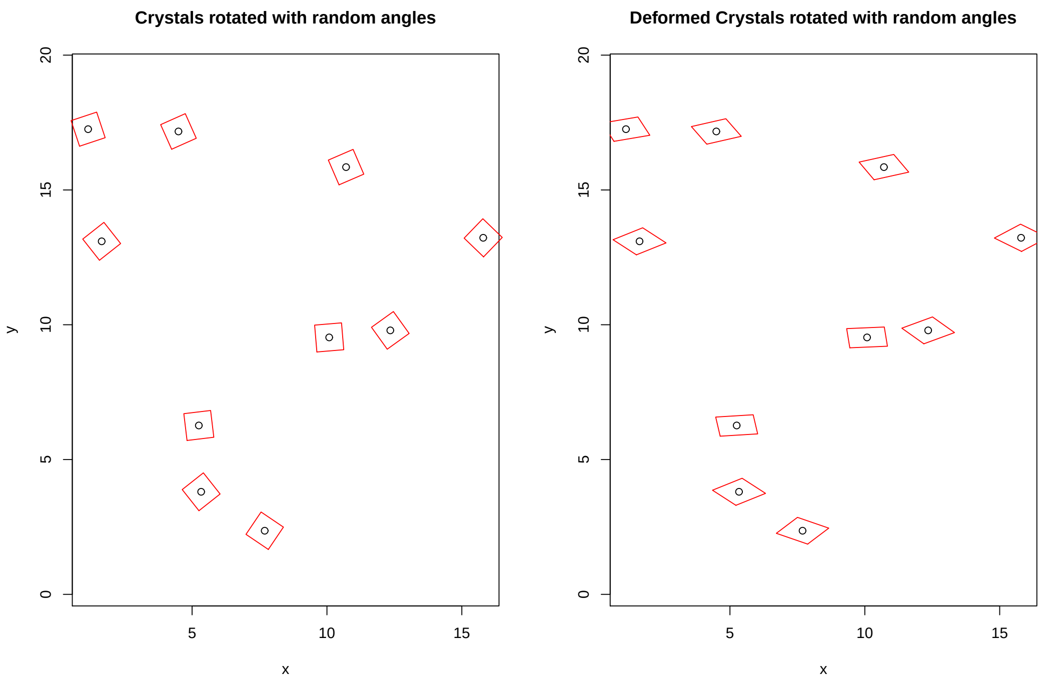

Figure 11: Given the handsample on the right we want to determine what shear has occured. We do not know, *a priori*, what the original rotations were for each crystal.

This may occur in sedimentary situations where gravity organizes crystals or where a constant temperature gradient controls crystalization.

Next we perform the polar decomposition on each inversion matrix:

```
> for (i in 1:n) {
+     ApU = PolarDecomp(DefList[[i]]$def)
+     DefList[[i]]$P = ApU$P
+     DefList[[i]]$U = ApU$U
+ }
```

Each Crystal has a different inverted resultant matrix - i.e. the matrix required to transform the observed crystals to the non-rotated, undeformed squares is different in each case. Even though the same deformation was applied to each crystal, the initial conditions were different for each object.

For each crystal, the polar decomposition of the inverted matrix shows that the $\boldsymbol{P}$ matrix is the same for all the crystals, even though the $\boldsymbol{U}$ matrix differ. Note that the $\boldsymbol{P}$ is the synthetic shear matrix originally set to deform the rotated crystals. Each of the $\boldsymbol{U}$ matrices correspond to the associated rotations applied when setting up the data. The inversions for the shear and rotations have provided a simple means to extract the critical information from the rock using a standard set of R-commands.

As an example, consider crystal sample number 3:

```
[1] "organism number 3"


[1] "rotation matrix (input):"


           [,1]      [,2] [,3]
[1,]  0.9489480 0.3154326    0
[2,] -0.3154326 0.9489480    0
[3,]  0.0000000 0.0000000    1


[1] "Inverted rotation result (derived):"


           [,1]      [,2]
[1,]  0.9489480 0.3154326
[2,] -0.3154326 0.9489480
```

Note that the known input rotation matrix and the derived matrix via inversion are the same. This approach may provide a way to extract deformation and rotation from a hand sample in the field if some simple assumptions can be made about the nature of the original deposition.

6 Fry's Method

An alternative method for extracting strain information from a hand sample is to consider the spacing between samples, given that small objects, when originally deposited have a finite volume and a similar inital shape. A geological example might include fossil micro-organisms or sorted grains in sand deposits.

A method for determining the strain applied to a conglomerate of small bodies cemented into the rock was developed by Norman Fry. Fry's method can be easily implemented and illustrated in R using the *geophysics* package.

Consider an example where a handsample consists of 200 reasonably defined small objects, say sand grains.

Fry's method consists of tabulating the relationship of each point to the others in the data set. The outline of this simple method involves: Each point is centered and the remaining points are plotted relative to this point keeping the original orientations.

Applying Fry's method in R we use the dofry routine to calcualte the image and plotfry to display.

```
> FF = dofry(DAT$x, DAT$y)
> plotfry(FF, dis = 30)
```

dev.off()

Note in Figure 13 that there is a central, nearly circular region that is devoid of points. The blank circle represents information about the spacing of the particles in original hand sample. Since the particles are randomly distributed, the blank spot is nearly circular, i.e. there is no prefered directionality to the spacing.

Next we deformed the hand sample with a given shearing mechanism, in this case simple shear.

```
> shr = 1.2
> simpleshear = matrix(c(1, shr, 0, 1), ncol = 2)
> x = DAT$x
> y = DAT$y
> APTS = cbind(x - mean(x), y - mean(y))
> NP = APTS %*% simpleshear
```

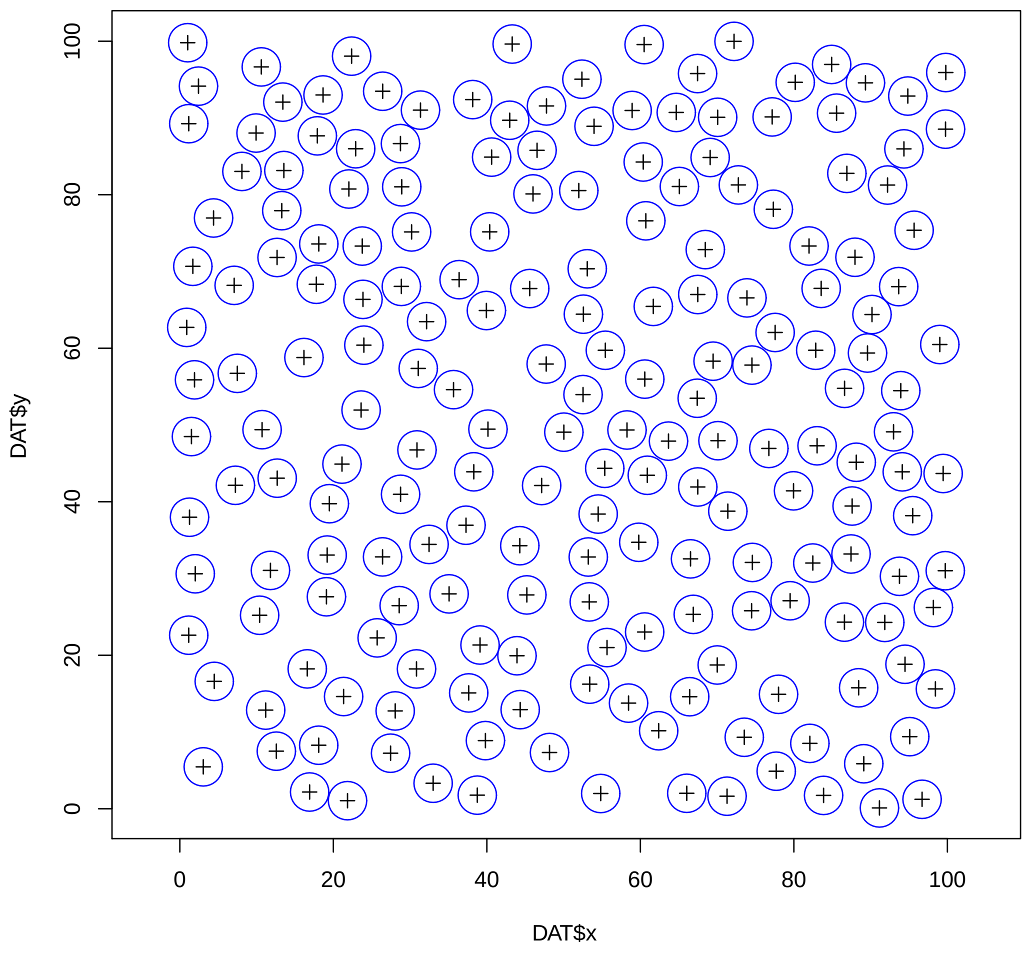

Figure 12: Hand Sample of 200 randomly distributed sand grains cemented in glass matrix. the grains cannot overlap. The center of each grain is indicated by a cross symbol.

```
> postscript(file = "FRY2.eps", width = 12, height = 8, paper = "special",
+     horizontal = FALSE, onefile = TRUE, print.it = FALSE)
> par(mfrow = c(1, 2))
> plot(NP[, 1], NP[, 2], asp = 1, type = "p", ann = FALSE, axes = FALSE)
> box()
```

```
> FF = dofry(NP[, 1], NP[, 2])
> AF = plotfry(FF, 20)
```

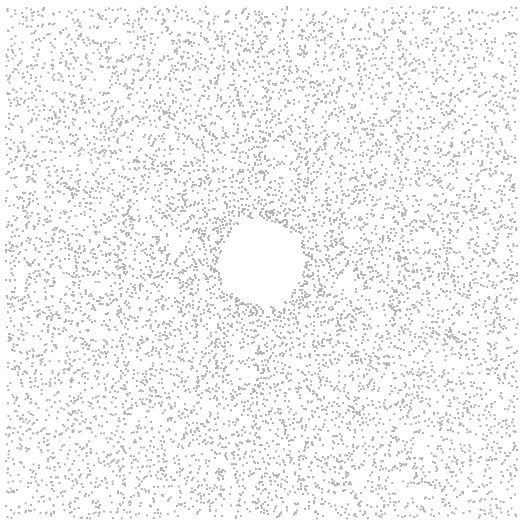

Figure 13: Fry method applied to random sample given in Figure 12.

```
> Z = xtractlip(AF)
> lines(predict(Z$hull), col = "red")
> title(format(shr))
```

It may seem obvious in the previous example that the shearing produced the output Fry diagram, because of the way the points were distributed after shearing. To illustrate that the shape of the input data does not influence the outcome of Fry's analysis we show three examples, randomly generated, where the data analyzed were selected from a circular subset of the artificially deformed data set.

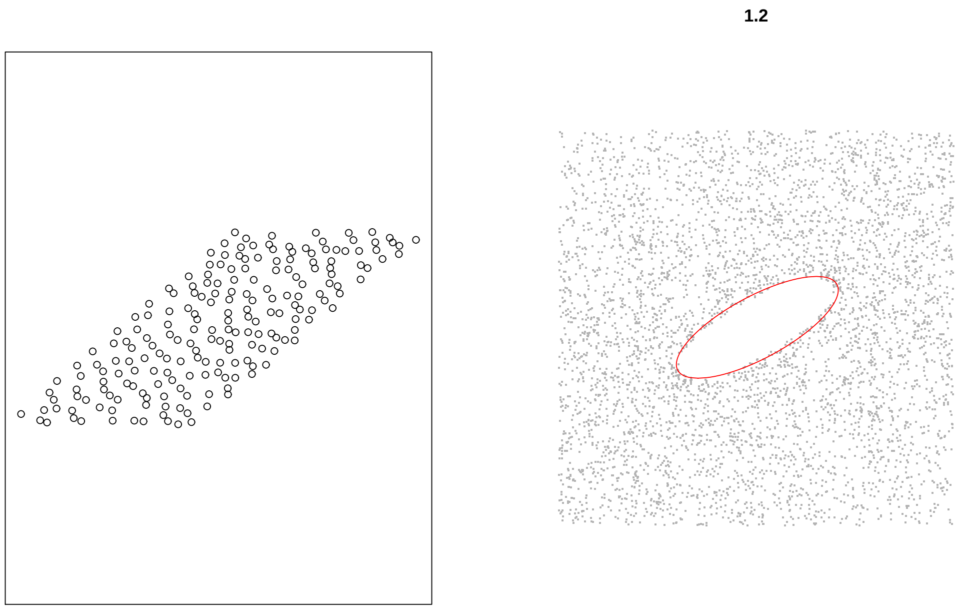

Figure 14: deformed hand sample

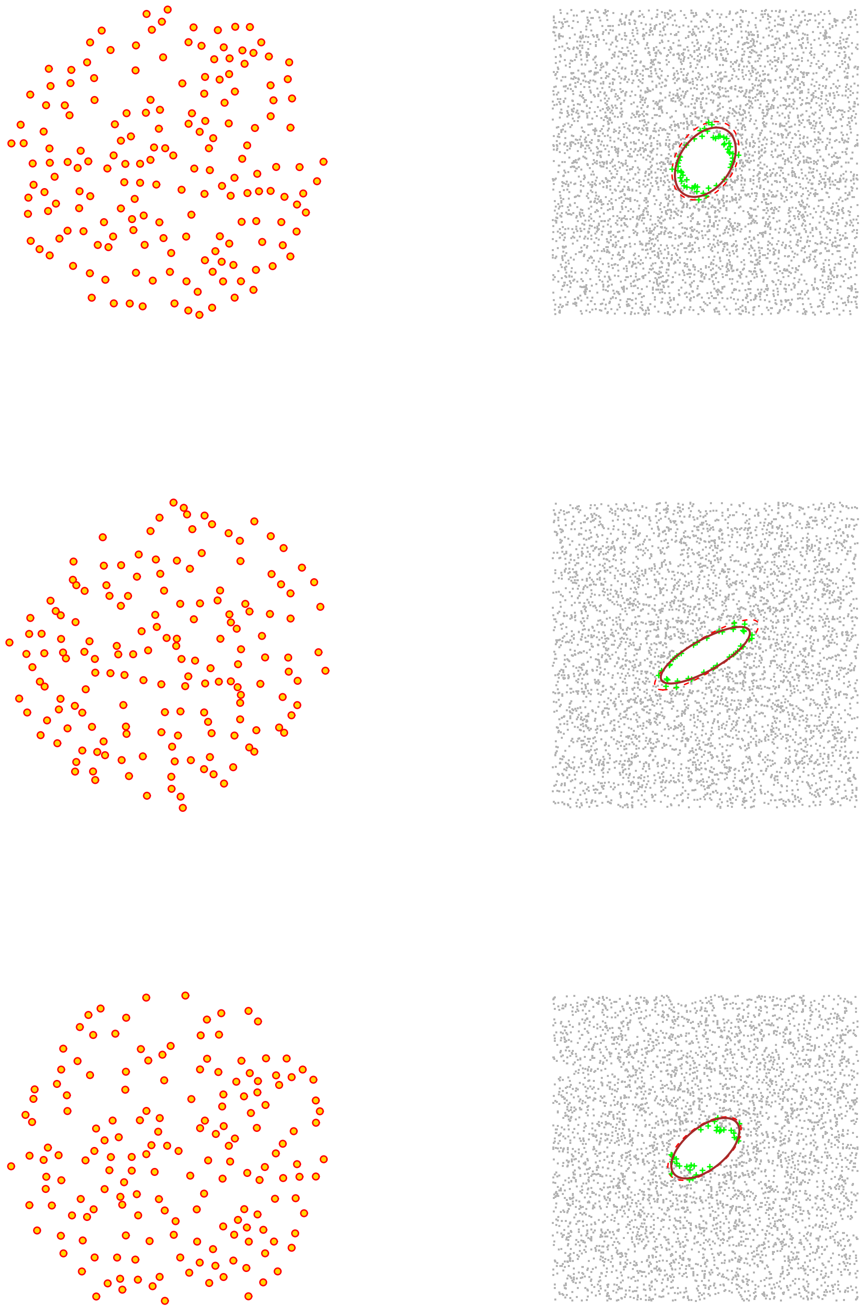

Figure 15: Several Fry diagrams showing different stresses. In these cases, nearly circular samples were extracted from a square distributions following deformation to illustrate that the overall shape of the sample does not have an influence on the Fry method.