# Backscatter harmonization using bulkshift() in R

This function is hosted permanently at https://github.com/benjaminmisiuk/bulkshift. Any changes, bugfixes, or updates will be available there.

## Setup and arguments

Source the function and check the arguments.

```
source("bulkshift.R")
args(bulkshift)
```

The args() call returns the following:

```
function (shift, target, preds = NULL, shift.method = "lm", mosaic = FALSE,
   save.data = FALSE, err.plots = TRUE, dist.plots = FALSE,
   sample = FALSE, samp.size = 0.25, family = gaussian, link = "identity",
   tc = 2, lr = 0.01, bf = 0.5, verb = FALSE)
```

The primary arguments from above are:

| | |
|---|---|
| shift | The backscatter raster being corrected. |
| target | The target backscatter dataset being referenced. |
| preds | Vector of optional additional predictors used for the bulk shift. |
| shift.method | The modelling method used; one of “mean”, “lm”, or “brt”. |
| mosaic | Output the corrected, harmonized backscatter mosaic? |
| save.data | Save and output the data frame used for bulk shift calculations? |
| err.plots | Output 2d (from bivariate models) or 3d (from multivariate models) error plots? |
| dist.plots | Output distribution (CDF and PDF) plots for shifted data? |
| sample | Subsample the area of overlap for large datasets? |
| samp.size | If sample=TRUE, what proportion of cells to sample between 0 and 1. |

When shift.method="lm", the following arguments are passed to glm():

| | |
|---|---|
| family | The error distribution. |
| link | The link function. |

When shift.method="brt", the following arguments are passed to gbm.step() (see the dismo package for details):

| | |
|---|---|
| tc | Tree complexity. |
| lr | Learning rate. |
| bf | Bag fraction. |
| verb | Trace the BRT progress? |

The arguments and code for the function can be viewed by calling:

```
View(bulkshift)
```

## Importing raster data

The first step is to import the raster backscatter data to R. This is done using the raster package.

```
library(raster)

#set the working directory to where the raster files are stored
setwd("C:/Documents/bulkshift_files")

#read in the shift and target layers
bedford_400 <- raster("bedford_400kHz")
bedford_200 <- raster("bedford_200kHz")
```

Before calling the function, ensure that the rasters have the same projection.

```
projection(bedford_400)
```

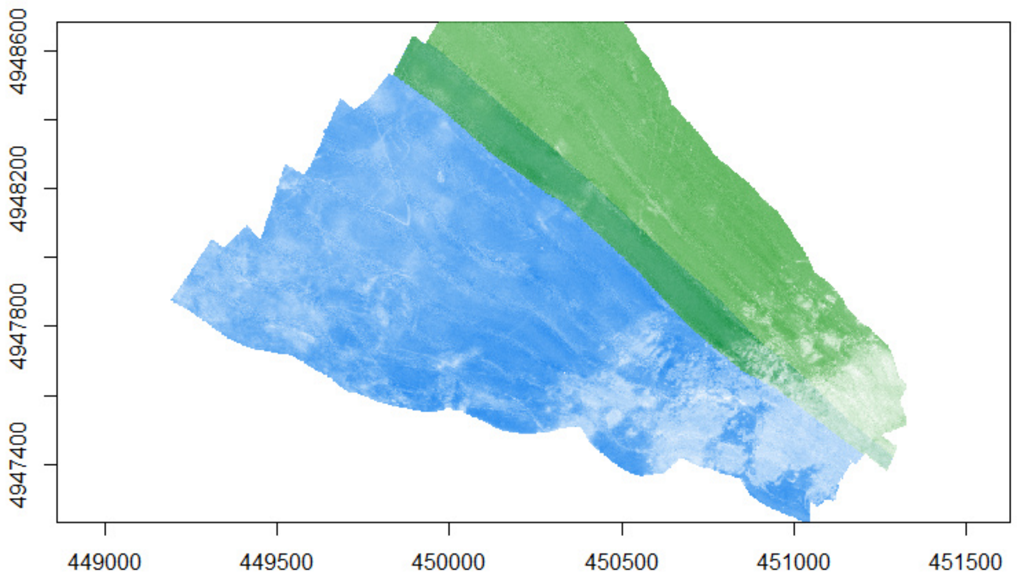
```
"+proj=utm +zone=20 +datum=WGS84 +units=m +no_defs +ellps=WGS84 +towgs84=0,0,0"
```

```
#check if the projections match
projection(bedford_400)==projection(bedford_200)
```

```
TRUE
```

It is best practice to ensure that the backscatters rasters have the same resolution before proceeding, and that they are properly aligned. If the resolutions do not match though, or if the rasters are unaligned, the function will automatically resample them to match. We can view the rasters before running the bulk shift to ensure that they overlap.

```
plot(bedford_200, col=colorRampPalette(c("dodgerblue2", "white"))(100), legend=FALSE)

plot(bedford_400, col=colorRampPalette(c("green4", "white"))(100), alpha=0.75, add=TRUE,
legend=FALSE)
```



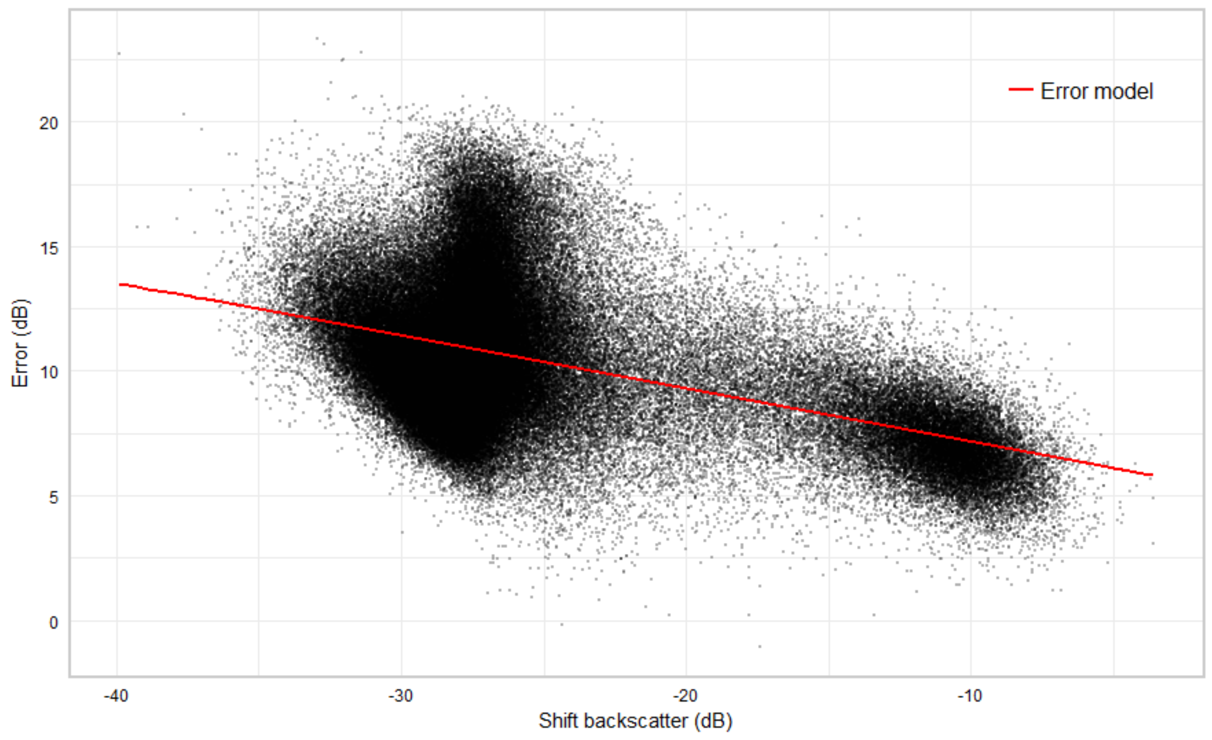
## An example using default arguments

Findings suggest that the default arguments for bulkshift() are generally the most transferable of the methods included in the function. Linear regression will be used by default. The function checks for required packages and installs any that are not found; this can take a few minutes, but only occurs on the first run.

```
bulk.shift <- bulkshift(bedford_400, bedford_200)
```

```
Loading required package: rgdal
Loading required package: dismo
Loading required package: ggplot2

Aligning rasters...
Fitting simple regression...
Predicting new raster...
Plotting error...

Complete. List contains 1: the shifted raster, 2: the fitted model, 3: bulk shift statistics

Fitted MAE = 1.952222 ; K-S statistic D = 0.1813082
```

The second to last line above describes the function output. The last line gives the evaluation statistics. Because err.plots=TRUE by default, the function has returned a plot of the error between backscatter datasets as function of the shift layer (400 kHz), including the fitted error model. In other words, the red regression line in the following plot represents the values that are applied to a given dB level in the 400 kHz dataset in an attempt to match the 200 kHz values.

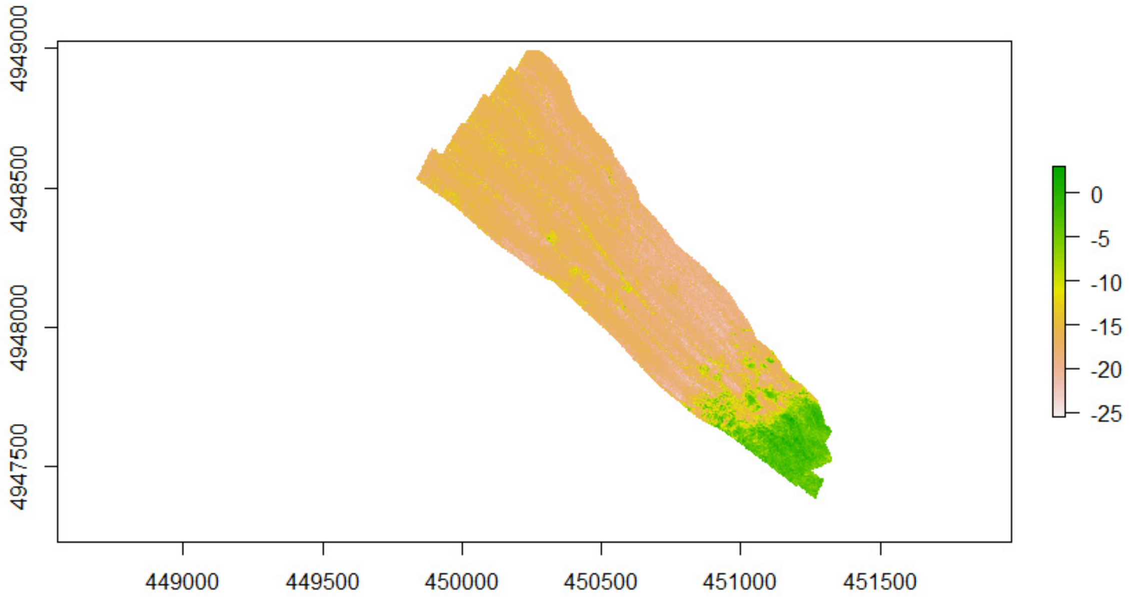


Next, we can access the function outputs. They are a list comprising three elements.

```
names(bulk.shift)

"shifted"   "model"    "fit.stats"
```

We can access each element of the list. For example, check and plot the shifted raster, check the model parameters, and investigate the evaluation statistics:

```
#check the raster
bulk.shift$shifted

class          :  RasterLayer
dimensions :  1794, 2142, 3842748  (nrow, ncol, ncell)
resolution   :  1, 1  (x, y)
extent         :  449190, 451332, 4947232, 4949026  (xmin, xmax, ymin, ymax)
coord. Ref.  :  +proj=utm +zone=20 +datum=WGS84 +units=m +no_defs +ellps=WGS84   +towgs84=0,0,0
data source :  in memory
names         :  layer
values         :  -32.72094, 4.819676  (min, max)

#plot the raster
plot(bulk.shift$shifted)
```



```
#check model parameters
bulk.shift$model

Call:  glm(formula = err ~ shift, family = family(link = link), data = back_data)

Coefficients:
(Intercept)        shift
     5.0406      -0.2126

Degrees of Freedom: 180763 Total (i.e. Null);  180762 Residual
Null Deviance:      1445000
Residual Deviance: 1133000      AIC: 844700

#check evaluation statistics
bulk.shift$fit.stats

      MAE         D
1.9522219 0.1813082
```

We can then export the corrected backscatter layer to the working directory for use in other GIS applications.

```
writeRaster(bulk.shift$shifted, filename = "bedford_400_corrected", format="GTiff")
```

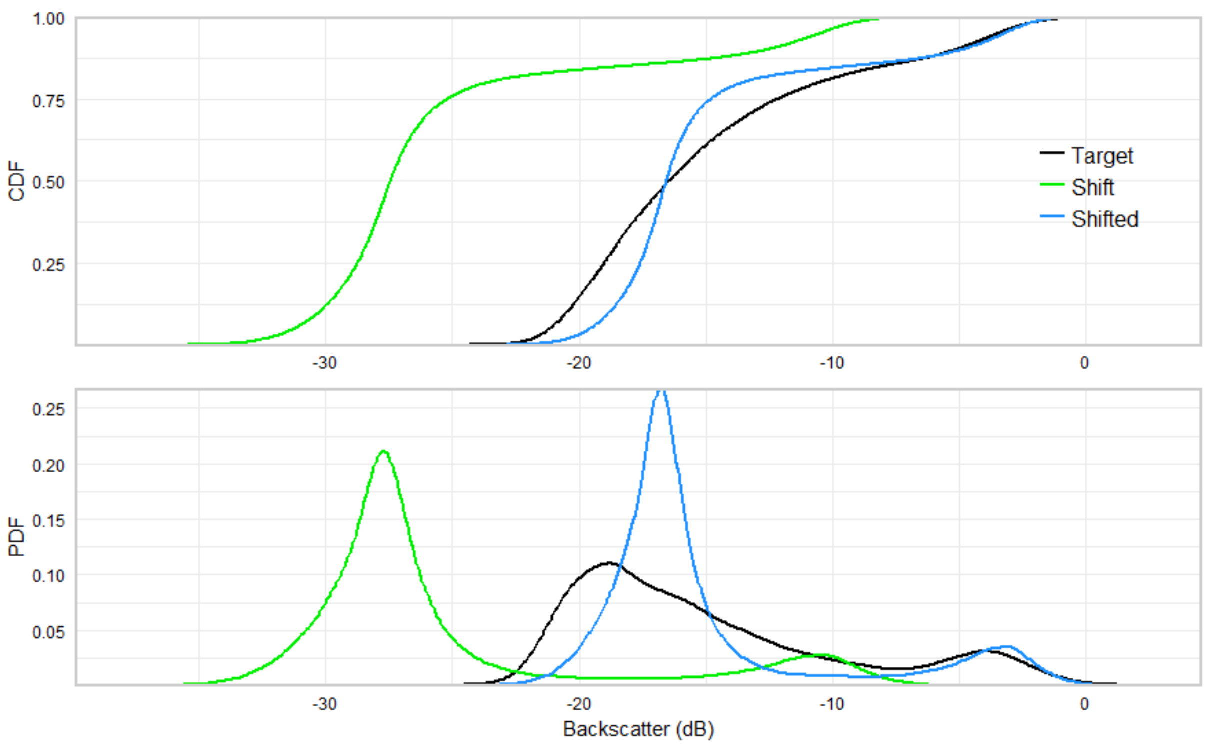
## Additional options

We can choose to output additional information on the bulk shift, and also mosaic the corrected layer with the target to produce a single harmonized dataset.

```
bulk.shift <- bulkshift(bedford_400, bedford_200, mosaic=TRUE, save.data=TRUE, dist.plots=TRUE)
```

```
Aligning rasters...
Fitting simple regression...
Predicting new raster...
Plotting error...

Loading required package: reshape2
Loading required package: cowplot

Creating harmonized mosaic...

Complete. List contains 1: the shifted raster, 2: the harmonized mosaic, 3: the fitted model, 4: bulk shift
statistics, 5: training data and predictions

Fitted MAE = 1.952222 ; K-S statistic D = 0.1813082
```

Because we set dist.plots=TRUE, the output now also includes plots of the probability density function (PDF) and cumulative distribution function (CDF) for the shift, shifted, and target datasets. These show that the shifted distributions match the target more closely.

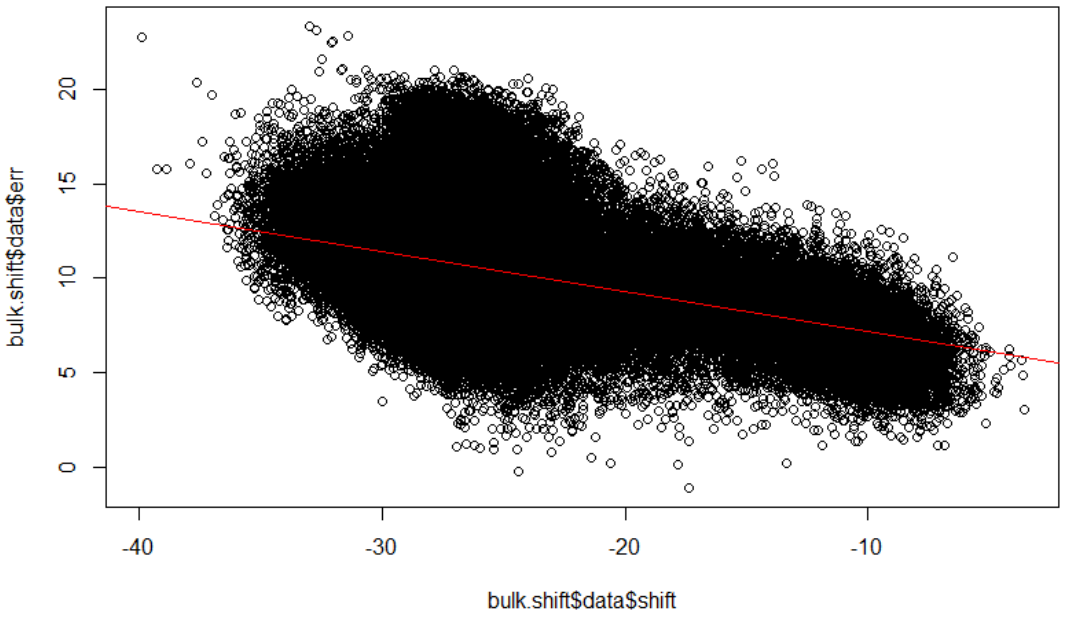


We have also chosen to save the modelling data frame (save.data=TRUE), which contains observations of the target and shift datasets, and the error between them, which is used to create the model. The last column contains the shifted values. These data could be used to set up an independent evaluation, conduct further modelling and correction, or produce custom plots.

```
head(bulk.shift$data)
```

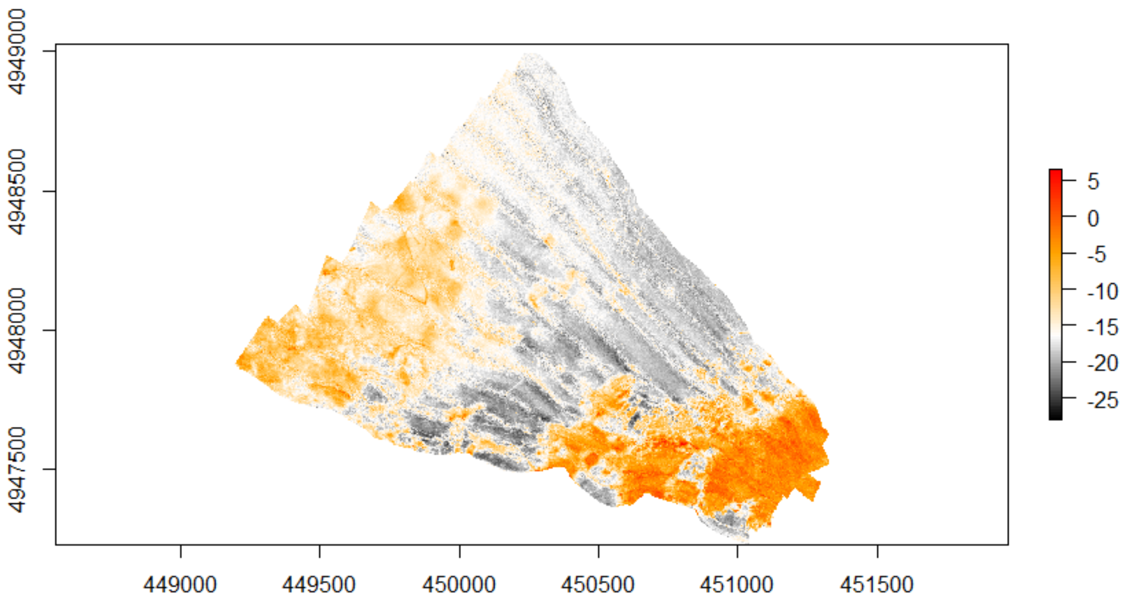
```
          target      shift  dummy      err    shifted
74515 -10.817500 -28.55511      1 17.73761  -17.44239
74516  -9.890500 -27.54522      1 17.65472  -16.64725
76624 -12.931600 -28.36830      1 15.43670  -17.29531
76625 -11.401750 -27.96323      1 16.56148  -16.97638
76626  -9.996727 -27.09061      1 17.09389  -16.28932
78732 -13.106501 -27.73611      1 14.62961  -16.79755
```

```
plot(bulk.shift$data$shift, bulk.shift$data$err)
abline(bulk.shift$model, col="red")
```



We have also saved the harmonized mosaic, which can be plotted and exported.

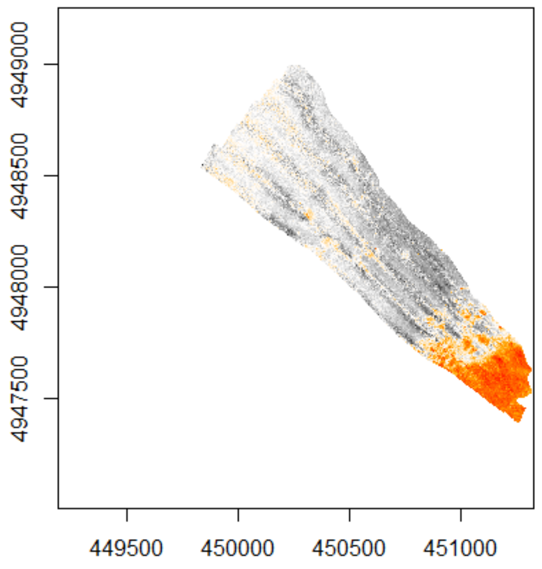
```
plot(bulk.shift$mosaic, col=colorRampPalette(c("black", "white", "orange", "red"))(100))
writeRaster(bulk.shift$mosaic, filename = "bedford_400_200_mosaic", format="GTiff")
```

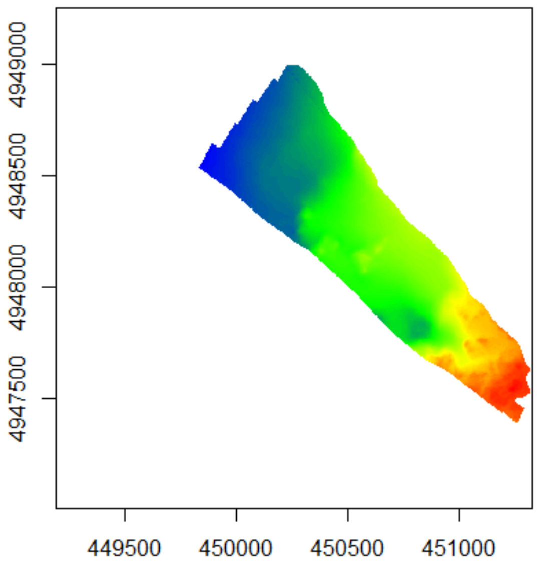


## Additional variables

If additional environmental variables that might help inform the correction are available, we can read them in as rasters.

```
#read in bathymetry raster from working directory
bathy <- raster("bathymetry")

#check the raster information and plot it
bathy

class       :  RasterLayer
dimensions :  1794, 2142, 3842748  (nrow, ncol, ncell)
resolution   :  1, 1  (x, y)
extent        :  449190, 451332, 4947232, 4949026  (xmin, xmax, ymin, ymax)
coord. Ref.  :  +proj=utm +zone=20 +datum=WGS84 +units=m +no_defs +ellps=WGS84   +towgs84=0,0,0
data source :  C:/Documents/bulkshift_files/bathymetry
names         :  bathymetry
values        :  -63.1, -13.87  (min, max)

layout(matrix(c(1,2), nrow=1, ncol=2))
plot(bedford_400, col=colorRampPalette(c("black", "white", "orange", "red"))(100), legend=FALSE)
plot(bathy, col=colorRampPalette(c("blue", "green", "yellow", "red"))(100), legend=FALSE)
```



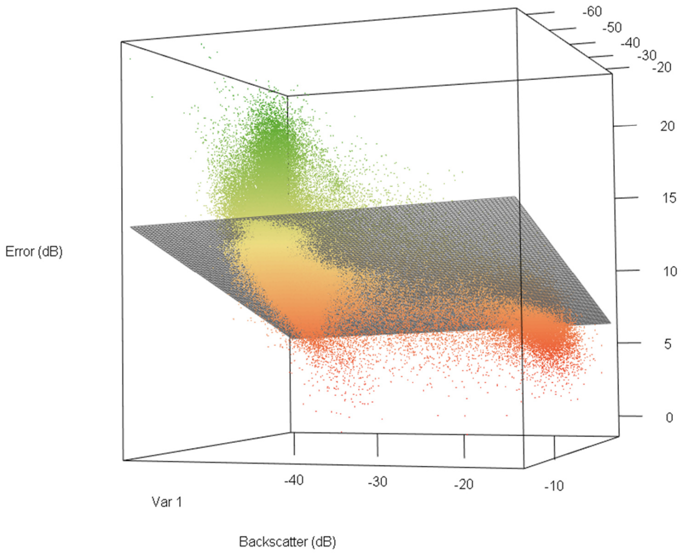


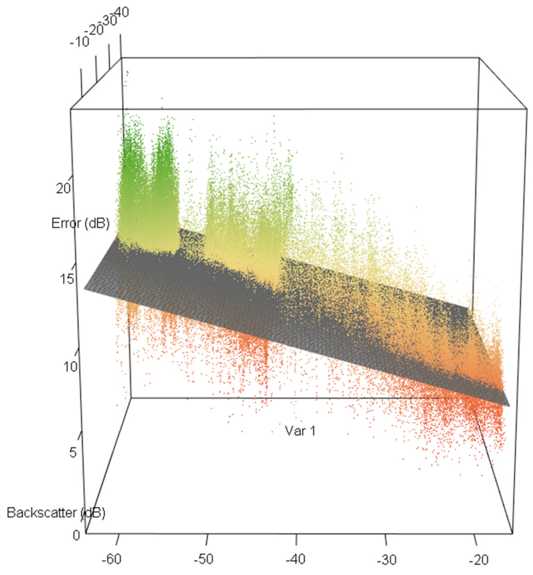
Both layers cover approximately the same area, so we can use bathymetry to help predict the bulk shift. **Note:** if the extents of additional covariates are less than the shift backscatter layer, all raster layers will automatically be masked and resampled to the layer of minimum extent by the bulkshift() function. We recommend ensuring that the extent of additional covariates is appropriate before running bulkshift().

```
bulk.shift <- bulkshift(bedford_400, bedford_200, preds=bathy)
```

```
Aligning rasters...
Fitting multiple regression...
Predicting new raster...
Plotting error...

Loading required package: plot3D
Loading required package: plot3Drgl
Loading required package: rgl

Complete. List contains 1: the shifted raster, 2: the fitted model, 3: bulk shift statistics

Fitted MAE = 1.765896 ; K-S statistic D = 0.111759
```

From the output, we can see that the fitted evaluation statistics (MAE and D) are better with the addition of bathymetry. Because we now have two predictors, the function will output an interactive 3d error plot using the plot3D, plot3Drgl, and rgl packages, with the multiple linear regression now represented as a plane. Here, we can explore the error in relation to the shift backscatter values and the bathymetry (called "Var 1" on the plot) using the mouse. Error plots are useful for exploring relationships between covariates and the error, and for diagnosing model fit, but they can be disabled using `err.plots=FALSE`.



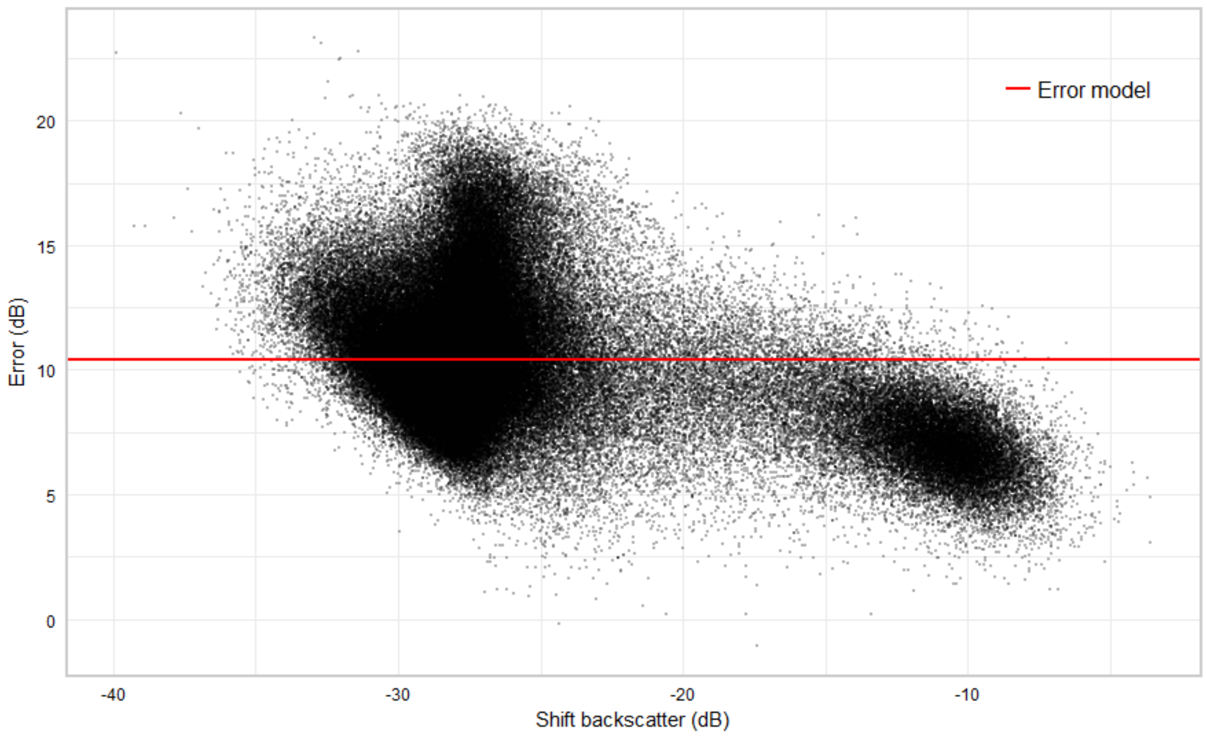


## Alternative modelling methods

The simplest method within the bulkshift() function is "mean", which adds the mean of the backscatter error to the shift layer. This is equivalent to an intercept-only model. It can run with additional covariates, but they will not influence the results.

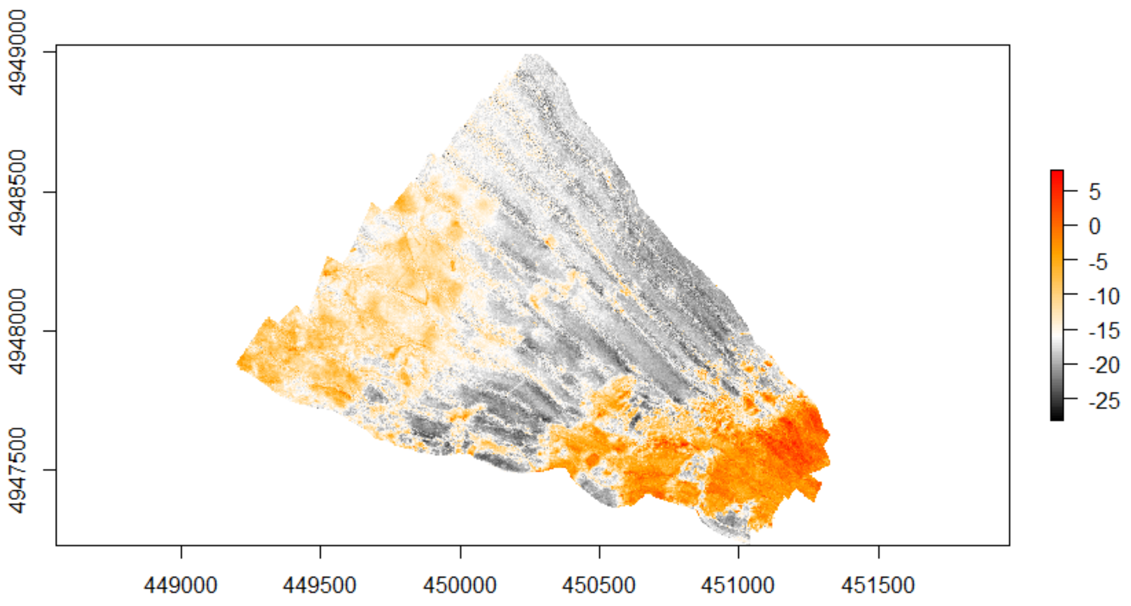
```
bulk.shift <- bulkshift(bedford_400, bedford_200, shift.method="mean", mosaic=TRUE,
dist.plots=FALSE)
```

```
Aligning rasters...
Shifting layer by the mean of the error...
Predicting new raster...
Plotting error...
Creating harmonized mosaic...

Complete. List contains 1: the shifted raster, 2: the harmonized mosaic, 3: the fitted model, 4: bulk shift
statistics

Fitted MAE = 2.238535 ; K-S statistic D = 0.1330243
```



The evaluation statistics are not as good than the regression methods. The error plot shows the mean value, which is simply added to all values in the shift layer. It is important to note that while the mean shift has the highest bias of all the methods included (the least flexible model fit), it also has the lowest variance, meaning that its performance on new data should be very similar to what is observed in model fitting. This method may require fewer samples (i.e., less overlap) than other methods, but may not produce as good results across the entire dataset. One potential approach is to first try the mean shift, observe the error plots and harmonized mosaic, then increase the model complexity as necessary using different methods, such as regression, and additional covariates.

```
dev.off() #may need to reset the graphics device after  previous plots
plot(bulk.shift$mosaic, col=colorRampPalette(c("black", "white", "orange", "red"))(100))
```



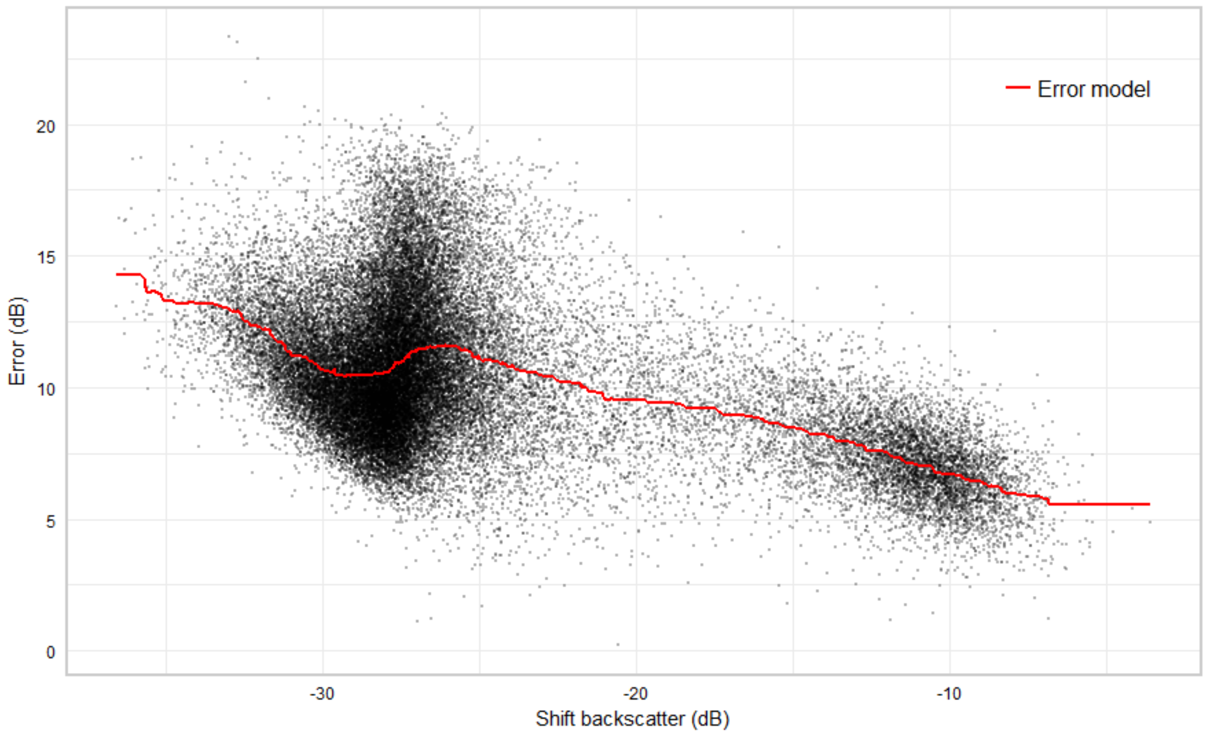
We can see visible differences between high dB values of the two datasets in the bottom right of the harmonized mosaic. The mean shift has not performed as well at these values as regression.

Conversely, boosted regression trees (BRT) are a highly flexible algorithm capable of modelling specific errors – they are implemented in the bulkshift() function using gbm.step() from the dismo package. The default hyperparameters for BRT in the bulkshift() function have been found to perform generally well (tc=2, lr=0.01, bf=0.5), but these can be tuned for individual applications to adjust the flexibility of the fitted model (i.e., the bias-variance tradeoff) – particularly the tree complexity (tc) and learning rate (lr). See the dismo package and Elith et al. (2008) for more details on these. BRT can take a long time to run with large datasets. We can subsample the dataset to speed up processing using sample=TRUE and samp.size=0.25; verb=TRUE tells the function to report the BRT progress.

```
bulk.shift <- bulkshift(bedford_400, bedford_200, shift.method="brt", dist.plots=FALSE, sample=TRUE,
samp.size=0.25, verb=TRUE)
```

```
Aligning rasters...
Fitting BRT model...

Loading required namespace: gbm

 GBM STEP - version 2.9

Performing cross-validation optimisation of a boosted regression tree model
for err and using a family of gaussian
Using 45191 observations and 2 predictors
creating 10 initial models of 50 trees

 folds are unstratified
total mean deviance =  7.9967
```

```
tolerance is fixed at  0.008
ntrees resid. dev.
50    6.8344
now adding trees...
100   6.3687
150   6.1634
200   6.0622
250   6.0055
300   5.9715
350   5.9485
400   5.9329
450   5.9218
500   5.9143
…
fitting final gbm model with a fixed number of 900 trees for err

mean total deviance = 7.997
mean residual deviance = 5.879

estimated cv deviance = 5.899 ; se = 0.04

training data correlation = 0.515
cv correlation =  0.512 ; se = 0.004

elapsed time -  0.04 minutes

Predicting new raster...
Plotting error...
Complete. List contains 1: the shifted raster, 2: the fitted model, 3: bulk shift statistics

Fitted MAE = 1.871337 ; K-S statistic D = 0.1441216
There were 11 warnings (use warnings() to see them)
```

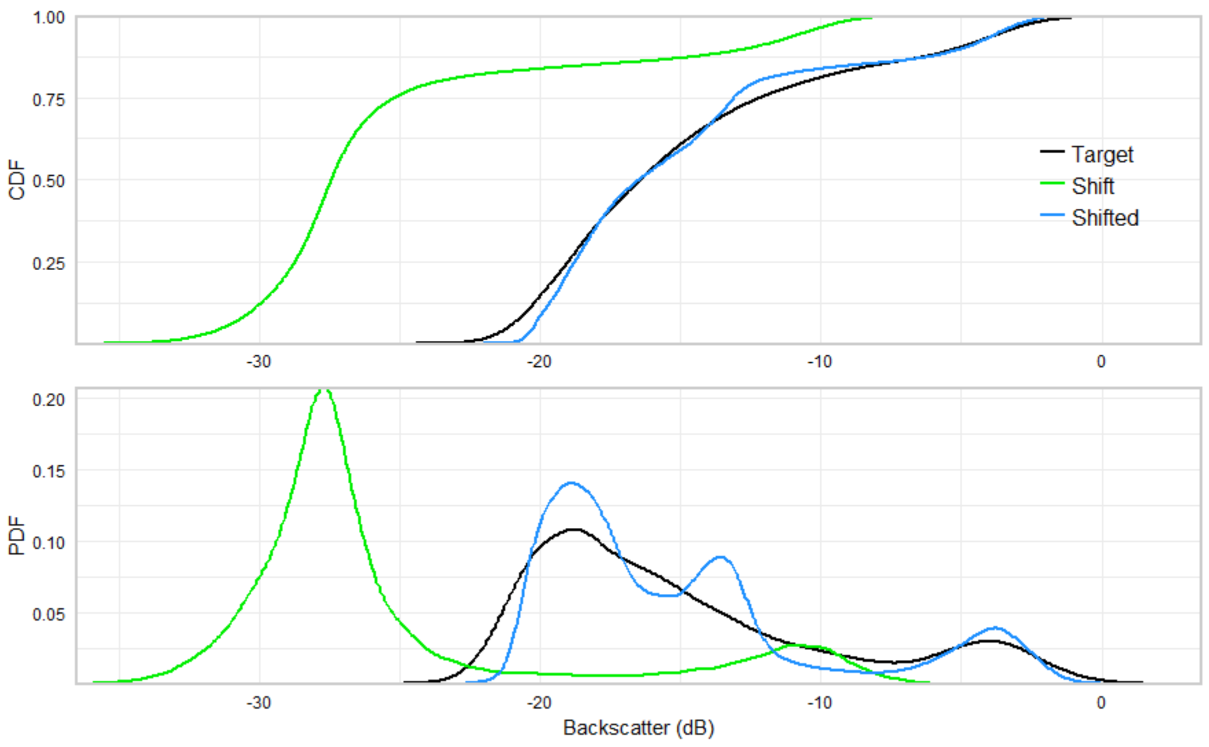


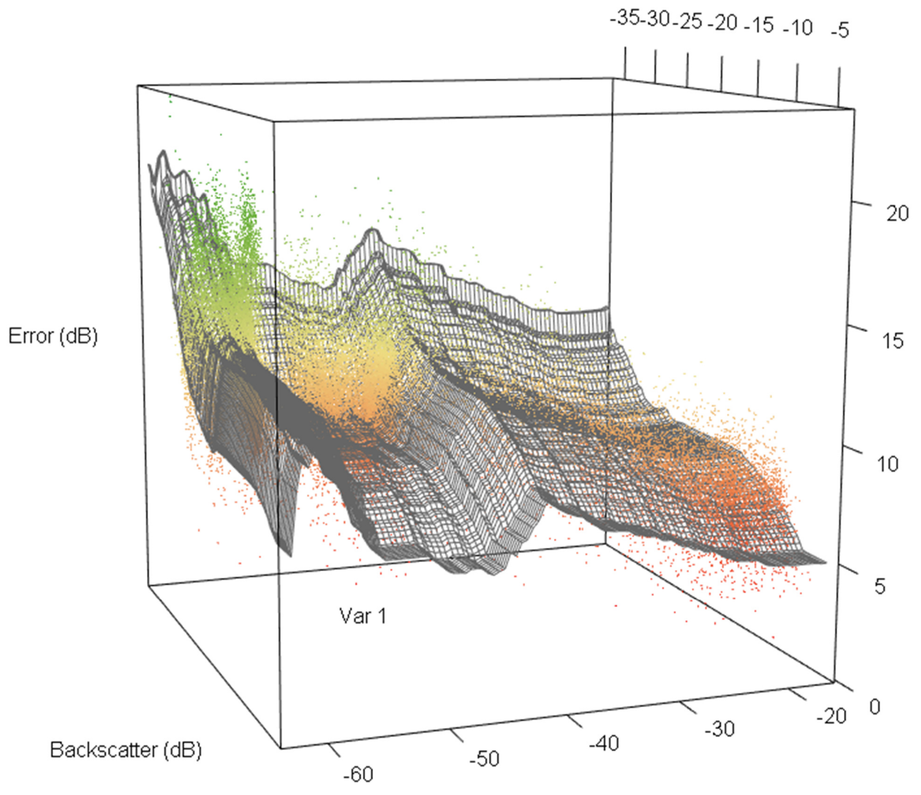
The output from gbm.step() is provided because we have specified verb=TRUE; it indicates the model progress. The left column is the current number of trees in the model, and the right is the residual deviance (these have been abridged for this tutorial with "…"). The model will stop fitting once the decrease in deviance falls below a threshold (determined by the tolerance). It is recommended to grow around 1000 trees (Elith et al., 2008), and this can be achieved by tuning tc and lr if the default results in fewer.

We can add bathymetry to the BRT model to see how it handles extra variables. This time, we will leave verb=FALSE (the default), but will set dist.plots=TRUE to see what effect the model has on the data distribution. **Note**: we can add as many predictors as desired, but error plots will not be produced when there are more than three. A potential approach for visualizing these is to set save.data=TRUE, then reduce the dimensionality (e.g., using PCA).

```
bulk.shift <- bulkshift(bedford_400, bedford_200, preds=c(bathy), shift.method="brt", mosaic=TRUE,
dist.plots=TRUE, sample=TRUE, samp.size=0.25)
```

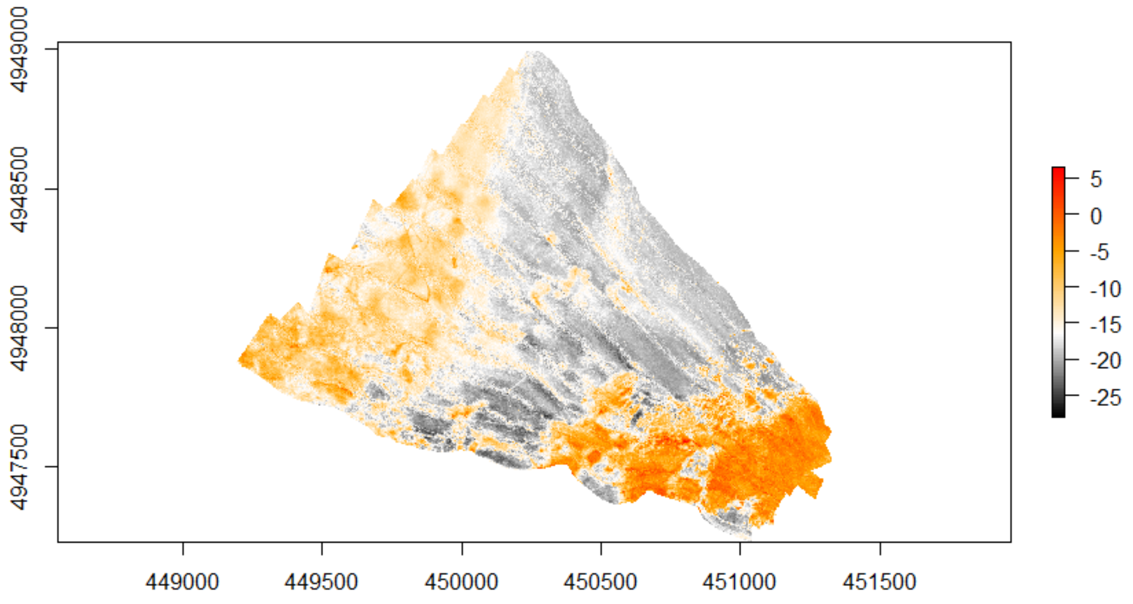
```
Aligning rasters...
Fitting BRT model...
Predicting new raster...
Plotting error...
Creating harmonized mosaic...
Complete. List contains 1: the shifted raster, 2: the harmonized mosaic, 3: the fitted model, 4: bulk shift
statistics

Fitted MAE = 1.429414 ; K-S statistic D = 0.07468301
```





The data distribution plots show that the shifted dataset values now match the target closely, and the 3d error plot shows that the model fit is complex. It is extremely important to note, though, that these represent the fitted model, and do not necessarily inform on its performance with new data. Our results have shown that the performance of flexible models on independent data was usually much worse than the fitted statistics indicated – flexible models such as these should be used with caution. The mosaic can sometimes provide further insight.

```
dev.off()
plot(bulk.shift$mosaic, col=colorRampPalette(c("black", "white", "orange", "red"))(100))
```



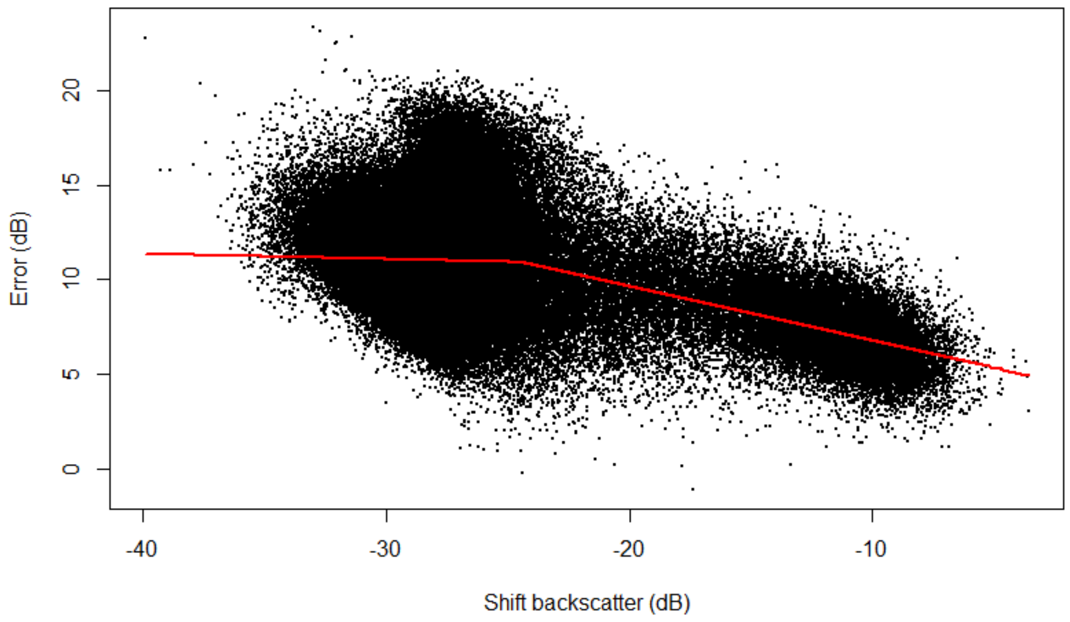
The model certainly seems to have reproduced the backscatter intensity of features that are visible in the target layer but there are now also features that seem suspect, such as the stripe of seabed in the eastern portion of the mosaic at around 16 dB. This feature was not visible in the original shift layer and may be an artefact caused by overfitting the bathymetry layer. Our simulations for this dataset suggest that multiple linear regression did, in fact, outperform BRT on withheld test data.

## Further model customization

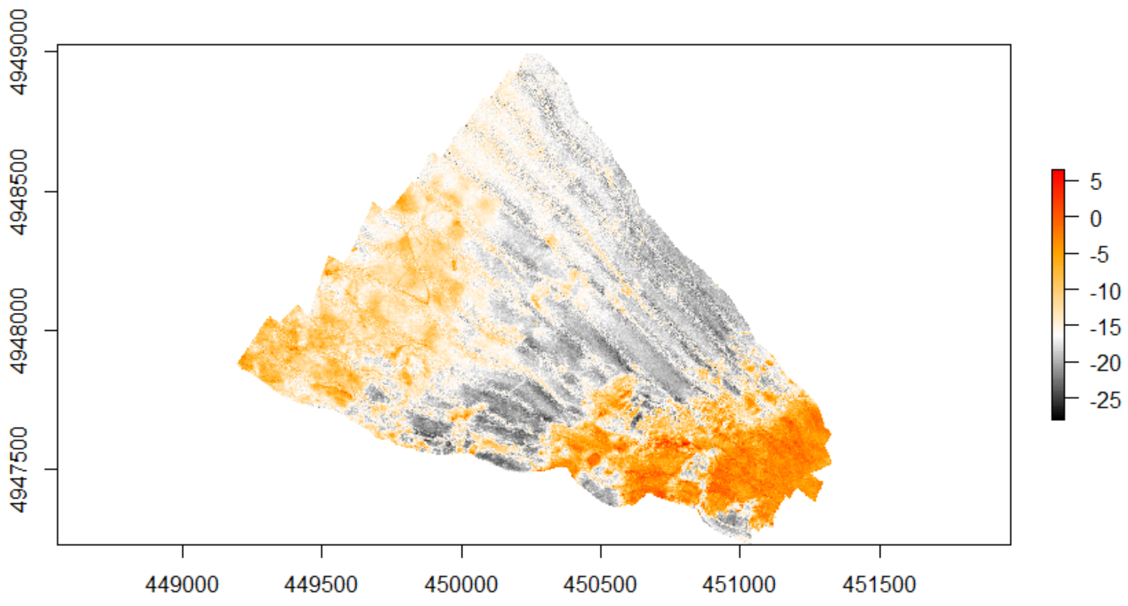
The bulkshift() function provides a streamlined framework for backscatter harmonization using several approaches tested in our paper, but many other statistical models could be used. When this is desirable, a useful approach is to first run a simple bulk shift using shift.method="mean", and save.data=TRUE, including any additional predictors, which will initially be ignored by the mean shift. This will align the rasters, extract the area of overlap, calculate the error between datasets, subsample if desired, and save the data. The saved data can then be used to test and apply other modelling approaches, and these can be compared to the mean shift, which acts as a benchmark.

```
bulk.shift <- bulkshift(bedford_400, bedford_200, preds=c(bathy), shift.method="mean",
save.data=TRUE)
```

```
Aligning rasters...
Shifting layer by the mean of the error...
Predicting new raster...
Plotting error...

Complete. List contains 1: the shifted raster, 2: the fitted model, 3: bulk shift statistics, 4. training data
and predictions

Fitted MAE = 2.238535 ; K-S statistic D = 0.1330243

head(bulk.shift$data)

                target      shift      bathy      err    shifted
 74515  -10.817500 -28.55511 -62.19810 17.73761 -18.13990
 74516   -9.890500 -27.54522 -62.17804 17.65472 -17.13001
 76624  -12.931600 -28.36830 -62.15000 15.43670 -17.95310
 76625  -11.401750 -27.96323 -62.15000 16.56148 -17.54803
 76626   -9.996727 -27.09061 -62.15000 17.09389 -16.67541
 78732  -13.106501 -27.73611 -62.21413 14.62961 -17.32091

#modelling the error using multivariate adaptive regression splines
library(earth)
mars.model <- earth(err ~ shift, data=bulk.shift$data)

#plot the fitted model values against the observed error for the shift layer
j <- order(bulk.shift$data$shift)

plot(bulk.shift$data$shift, bulk.shift$data$err, cex=0.25, pch=19, xlab="Shift backscatter (dB)",
ylab="Error (dB)")

lines(bulk.shift$data[j,2], mars.model$fitted.values[j], col="red", lwd=2)
```



```
#predict the error raster and add it to the shift layer
mars.corr <- (predict(bedford_400, mars.model)) + bedford_400

#align origin and extent of the corrected layer with the target then plot the mosaic
mars.align <- resample(mars.corr, extend(bedford_200, mars.corr), method="ngb")

plot(mosaic(mars.align, bedford_200, fun=mean), col=colorRampPalette(c("black", "white", "orange",
"red"))(100))
```



## References

Adler, D., Murdoch, D., et al. (2019). rgl: 3D Visualization Using OpenGL. R package version 0.100.19. https://CRAN.R-project.org/package=rgl

Elith, J., Leathwick, J. R., & Hastie, T. (2008). A working guide to boosted regression trees. Journal of Animal Ecology, 77(4), 802–813. https://doi.org/10.1111/j.1365-2656.2008.01390.x

Hijmans, R. J. (2019). raster: Geographic Data Analysis and Modeling. R package version 2.8-19. https://CRAN.R-project.org/package=raster

Hijmans, R. J., Phillips, S., Leathwick, J., & Elith, J. (2017). dismo: Species Distribution Modeling. R package version 1.1-4. https://CRAN.R-project.org/package=dismo

Soetaert, K. (2016). plot3Drgl: Plotting Multi-Dimensional Data - Using 'rgl'. R package version 1.0.1. https://CRAN.R-project.org/package=plot3Drgl

Soetaert, K. (2017). plot3D: Plotting Multi-Dimensional Data. R package version 1.1.1. https://CRAN.R-project.org/package=plot3D