Determine how to transform the target

Charlie Brummitt

Load the data

First, load in the data for Rpop (another name for what is called "absolute advantage" in the paper) by reading in the CSV file generated by the Jupyter notebook "Create_figures.ipynb":

```
Rpop = read.csv("../data_for_use_with_mgcv_in_R/Rpop__data_target__pca_2__target_is_difference_True.csv
```

Analysis

Create GAM model

First, we define the three GAMs used in the paper, with a cubic regression spline basis (a.k.a. CRS, `bs="cr"`) of rank 30. Rank 30 is sufficiently large so that the model can have high variance, but the smoothing parameter reduces the variance of the model.

```
rank = 30
basis = "cr"
cs_pc0 = gam(
  change_in_pc0 ~ (s(pc0_last_year, bs=basis, k=rank)
                  + s(pc1_last_year, bs=basis, k=rank)
                  + s(log10_gdp_per_capita_constant2010USD_last_year, bs=basis, k=rank)),
  data = Rpop)
cs_pc1 = gam(
  change_in_pc1 ~ (s(pc0_last_year, bs=basis, k=rank)
                  + s(pc1_last_year, bs=basis, k=rank)
                  + s(log10_gdp_per_capita_constant2010USD_last_year, bs=basis, k=rank)),
  data = Rpop)
cs_gdp = gam(
  change_in_log10_gdp_per_capita_constant2010USD ~ (
      s(pc0_last_year, bs=basis, k=rank)
      + s(pc1_last_year, bs=basis, k=rank)
      + s(log10_gdp_per_capita_constant2010USD_last_year, bs=basis, k=rank)),
  data = Rpop)
```

Analyze significance of terms

In the GAM for predicting the change in the score on the first principal component, we find that all three terms are significant (p-value < 1e-9):

```
summary(cs_pc0)
```

```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## change_in_pc0 ~ (s(pc0_last_year, bs = basis, k = rank) + s(pc1_last_year,
##     bs = basis, k = rank) + s(log10_gdp_per_capita_constant2010USD_last_year,
```

```
##     bs = basis, k = rank))
##
## Parametric coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.06020    0.01035   5.815 6.41e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##                                                     edf Ref.df      F
## s(pc0_last_year)                                 17.149 20.326 21.241
## s(pc1_last_year)                                  6.801  8.195  7.597
## s(log10_gdp_per_capita_constant2010USD_last_year)  3.948  5.024 12.488
##                                                   p-value
## s(pc0_last_year)                                  < 2e-16 ***
## s(pc1_last_year)                                 2.65e-10 ***
## s(log10_gdp_per_capita_constant2010USD_last_year) 4.14e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.0856   Deviance explained = 9.04%
## GCV = 0.57353  Scale est. = 0.57041   n = 5323
```

In the GAM for predicting the change in the score on the second principal component, we find that only the score on the first principal component is significant at the 5% level:

```
summary(cs_pc1)
```

```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## change_in_pc1 ~ (s(pc0_last_year, bs = basis, k = rank) + s(pc1_last_year,
##     bs = basis, k = rank) + s(log10_gdp_per_capita_constant2010USD_last_year,
##     bs = basis, k = rank))
##
## Parametric coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 0.004400   0.005766   0.763    0.445
##
## Approximate significance of smooth terms:
##                                                    edf Ref.df      F
## s(pc0_last_year)                                 7.197  8.963  1.855
## s(pc1_last_year)                                 7.273  8.714 23.678
## s(log10_gdp_per_capita_constant2010USD_last_year) 2.325  2.990  0.615
##                                                  p-value
## s(pc0_last_year)                                   0.054 .
## s(pc1_last_year)                                  <2e-16 ***
## s(log10_gdp_per_capita_constant2010USD_last_year)   0.610
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.0442   Deviance explained = 4.72%
## GCV = 0.17759  Scale est. = 0.177     n = 5323
```

In the GAM for predicting the change in the log-base-10 of per-capita incomes, we find that all three terms are significant at the 1% level:

```r
summary(cs_gdp)
```

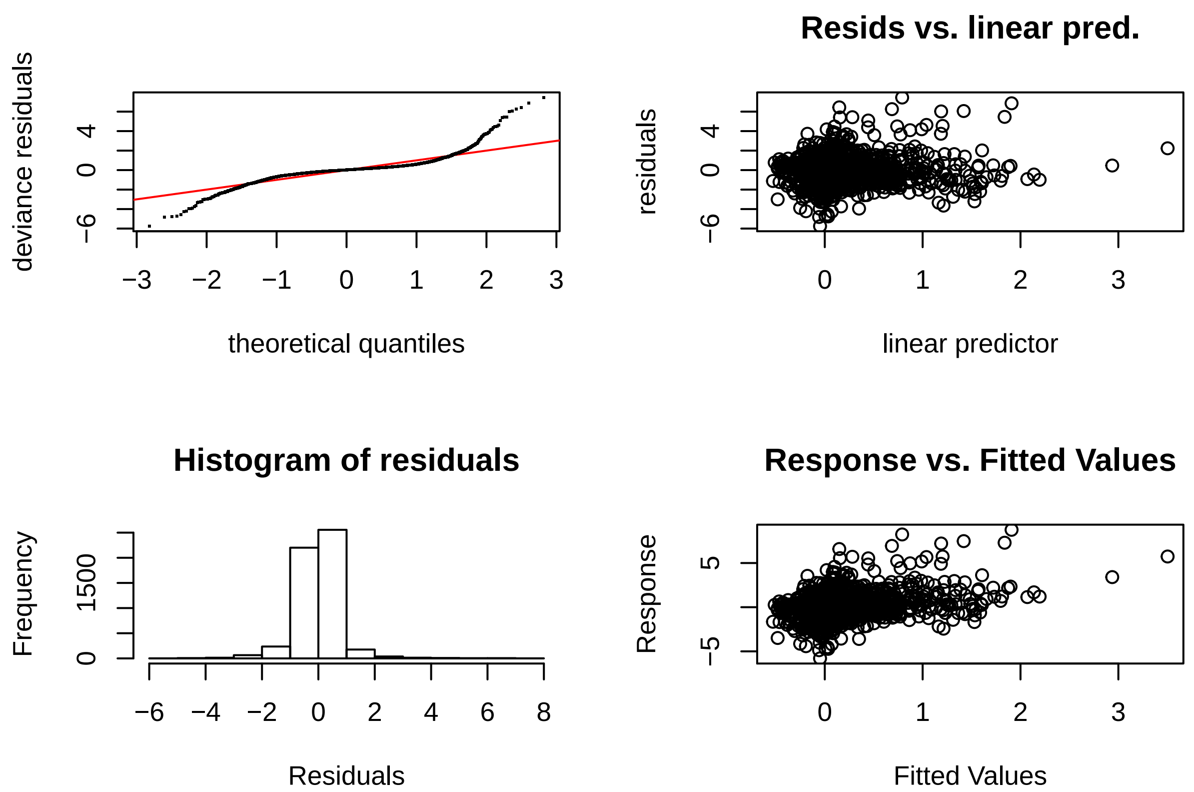
```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## change_in_log10_gdp_per_capita_constant2010USD ~ (s(pc0_last_year,
##     bs = basis, k = rank) + s(pc1_last_year, bs = basis, k = rank) +
##     s(log10_gdp_per_capita_constant2010USD_last_year, bs = basis,
##         k = rank))
##
## Parametric coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 0.0088723  0.0003276   27.09   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##                                                    edf Ref.df      F
## s(pc0_last_year)                                 7.667  9.534  9.695
## s(pc1_last_year)                                 6.764  8.152  2.868
## s(log10_gdp_per_capita_constant2010USD_last_year) 5.113  6.462 15.578
##                                                   p-value
## s(pc0_last_year)                                 2.22e-15 ***
## s(pc1_last_year)                                  0.00379 **
## s(log10_gdp_per_capita_constant2010USD_last_year)  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.0354   Deviance explained = 3.89%
## GCV = 0.00057332  Scale est. = 0.0005711  n = 5323
```

Check each of the GAMs

The mgcv package has a function gam.check that visualizes the residuals, a quantile-quantile plot, and more. In all three GAMs, the plots suggest that the tails of the residuals are heavy and that the "sholders" are narrow (compared to a normal distribution). These plots suggest that may want to apply (1/4)th root or square root to the response.

Check the GAM that predicts the change in the score on the first principal component

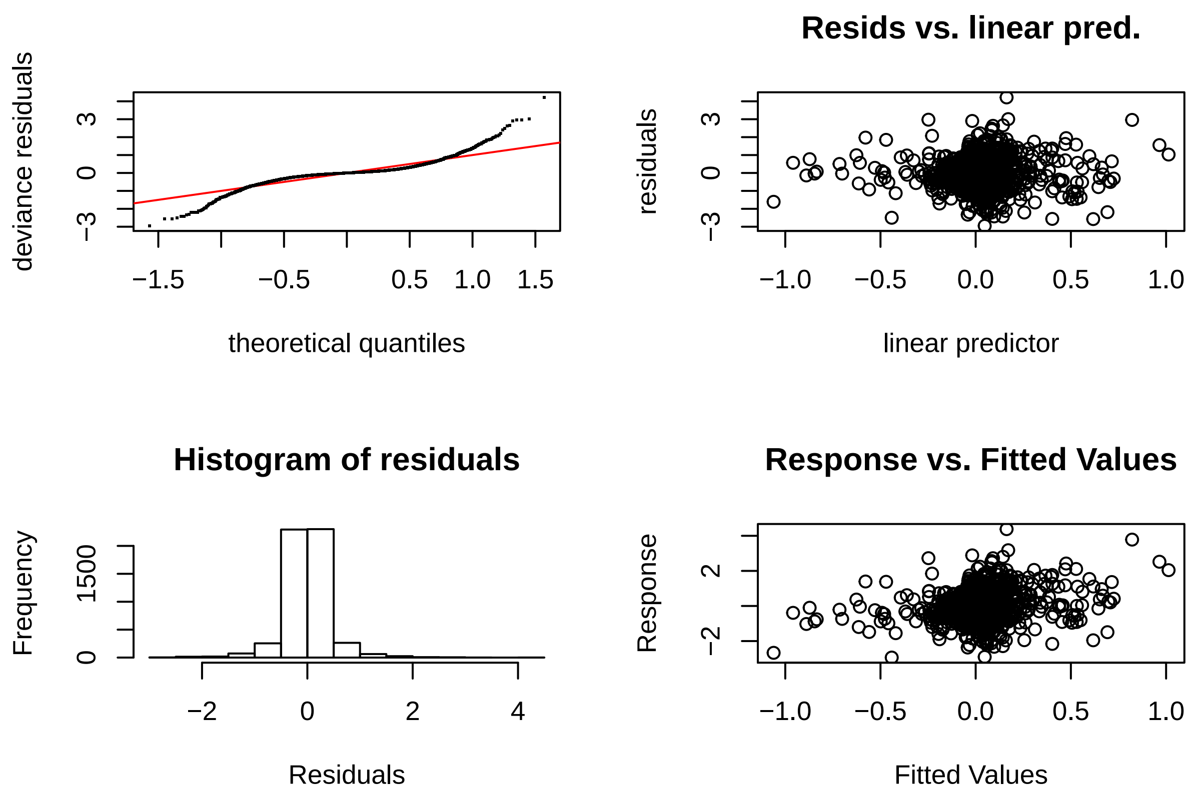
Notice in particular the deviation from the red line in the quantile-quantile plot (top-left plot), especially in the tails:




```
##
## Method: GCV   Optimizer: magic
## Smoothing parameter selection converged after 11 iterations.
## The RMS GCV score gradiant at convergence was 3.595723e-07 .
## The Hessian was positive definite.
## The estimated model rank was 88 (maximum possible: 88)
## Model rank =  88 / 88
##
## Basis dimension (k) checking results. Low p-value (k-index<1) may
## indicate that k is too low, especially if edf is close to k'.
##
##                                                k'    edf  k-index
## s(pc0_last_year)                           29.000 17.149   0.991
## s(pc1_last_year)                           29.000  6.801   1.007
## s(log10_gdp_per_capita_constant2010USD_last_year) 29.000  3.948   0.992
##                                            p-value
## s(pc0_last_year)                              0.22
## s(pc1_last_year)                              0.72
## s(log10_gdp_per_capita_constant2010USD_last_year)    0.29
```

Check the GAM that predicts the change in the score on the second principal component

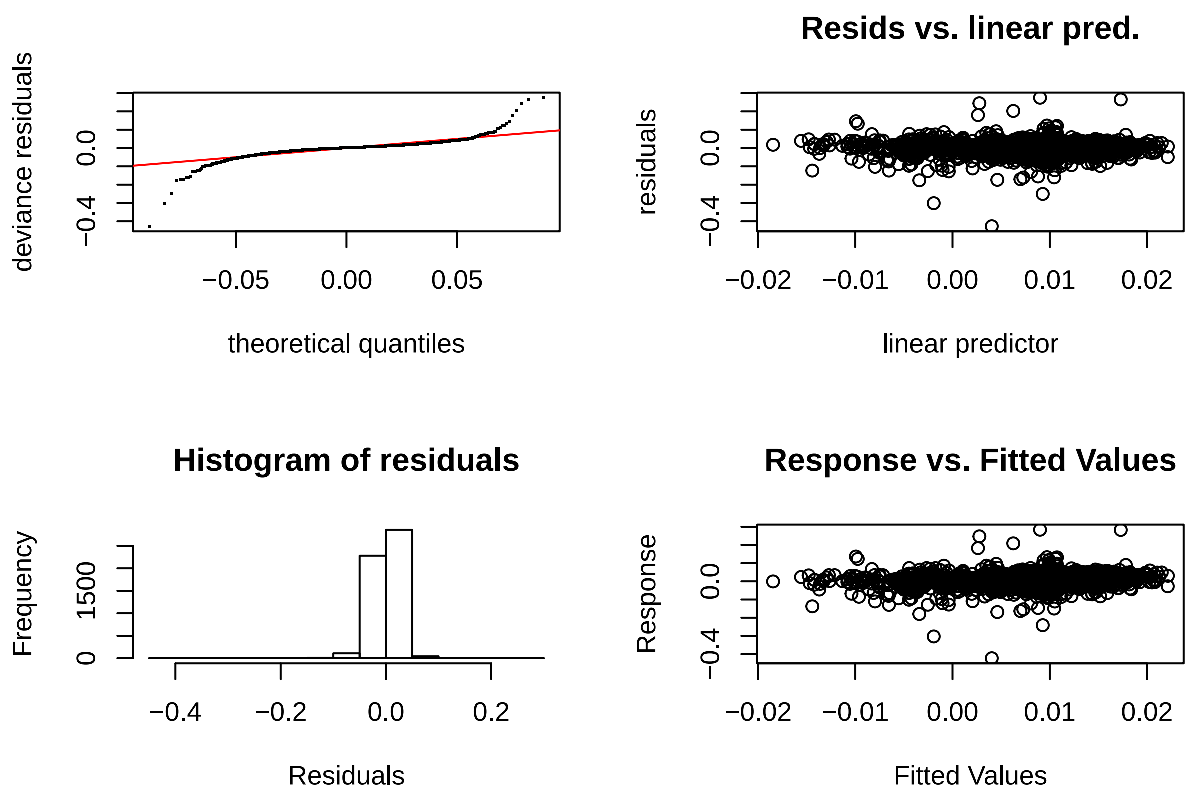
Again, the quantile-quantile curve (top-left plot) differs significantly from a straight line in the tails:




```
##
## Method: GCV   Optimizer: magic
## Smoothing parameter selection converged after 7 iterations.
## The RMS GCV score gradiant at convergence was 1.742752e-06 .
## The Hessian was positive definite.
## The estimated model rank was 88 (maximum possible: 88)
## Model rank =  88 / 88
##
## Basis dimension (k) checking results. Low p-value (k-index<1) may
## indicate that k is too low, especially if edf is close to k'.
##
##                                                 k'   edf k-index
## s(pc0_last_year)                             29.00  7.20    1.01
## s(pc1_last_year)                             29.00  7.27    1.01
## s(log10_gdp_per_capita_constant2010USD_last_year) 29.00  2.33    1.01
##                                              p-value
## s(pc0_last_year)                                0.74
## s(pc1_last_year)                                0.76
## s(log10_gdp_per_capita_constant2010USD_last_year)    0.75
```

Check the GAM that predicts the change in the log-base-10 of per-capita incomes

For the equation that predicts changes in (log) GDP per capita, the quantile-quantile curve (top-left plot) also differs significantly from a straight line in the tails:




```
##
## Method: GCV   Optimizer: magic
## Smoothing parameter selection converged after 8 iterations.
## The RMS GCV score gradiant at convergence was 5.287713e-08 .
## The Hessian was positive definite.
## The estimated model rank was 88 (maximum possible: 88)
## Model rank =  88 / 88
##
## Basis dimension (k) checking results. Low p-value (k-index<1) may
## indicate that k is too low, especially if edf is close to k'.
##
##                                                k'   edf k-index
## s(pc0_last_year)                           29.000 7.667   0.995
## s(pc1_last_year)                           29.000 6.764   0.996
## s(log10_gdp_per_capita_constant2010USD_last_year) 29.000 5.113   1.032
##                                            p-value
## s(pc0_last_year)                              0.36
## s(pc1_last_year)                              0.36
## s(log10_gdp_per_capita_constant2010USD_last_year)    0.96
```

Transform the target with square root while preserving the sign

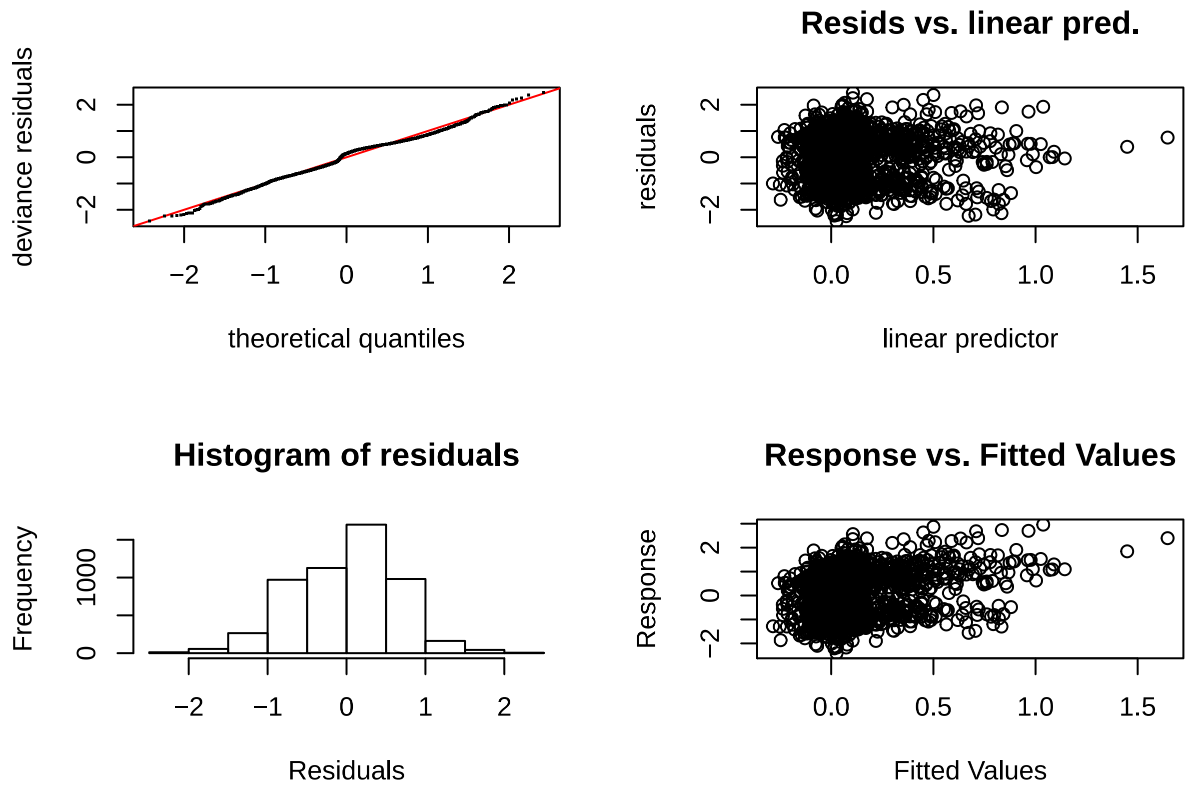
The results of `gam.check` above suggest that for all three GAMs we should try transforming the target to bring in the tails. Here we try the square root. To preserve the sign, we multiply the square root of the target by the `sign` of the target.

Transform the target for the GAM that predicts the change in the score on the first principal component

```
root = 0.5
cs_pc0_root = gam(abs(change_in_pc0)^root * sign(change_in_pc0) ~ s(pc0_last_year, bs=basis, k=rank) +
```

Now the quantile-quantile plot looks much more straight:

```
gam.check(cs_pc0_root)
```



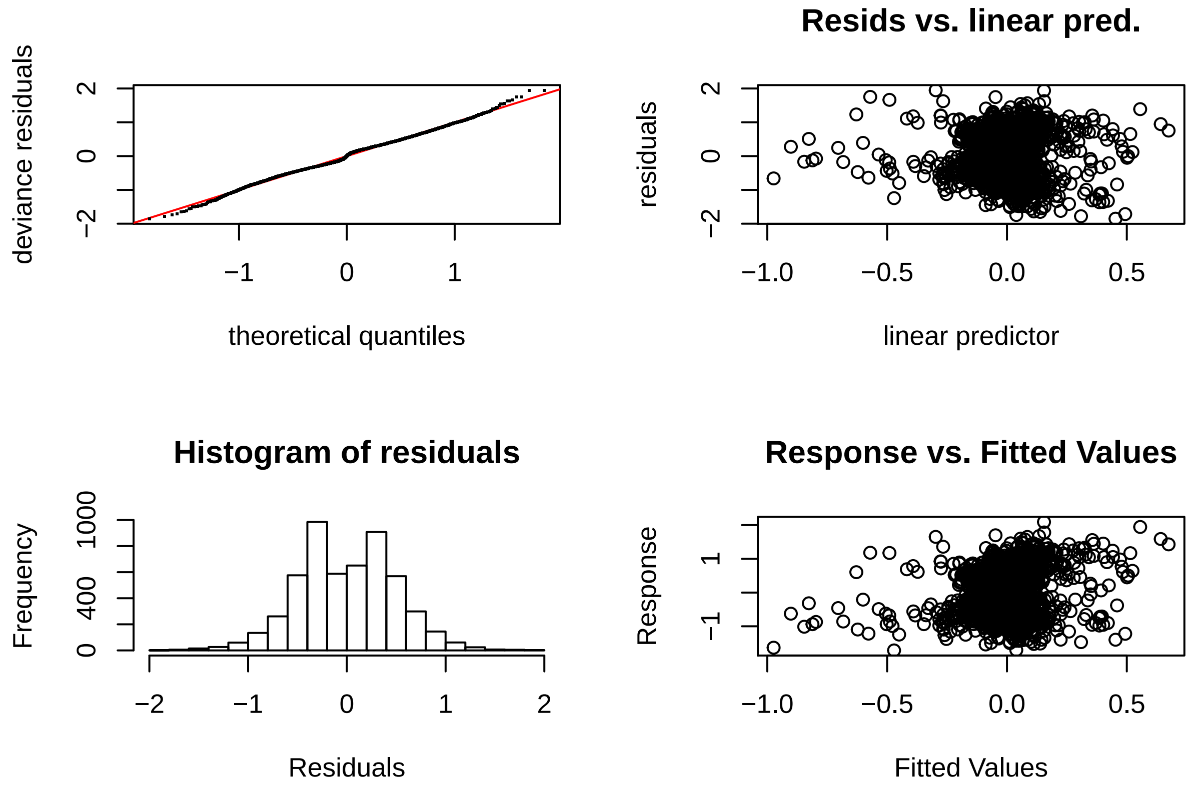

```
##
## Method: GCV   Optimizer: magic
## Smoothing parameter selection converged after 7 iterations.
## The RMS GCV score gradiant at convergence was 7.564506e-07 .
## The Hessian was positive definite.
## The estimated model rank was 88 (maximum possible: 88)
## Model rank =  88 / 88
##
## Basis dimension (k) checking results. Low p-value (k-index<1) may
## indicate that k is too low, especially if edf is close to k'.
##
##                                                    k'    edf k-index
## s(pc0_last_year)                               29.000 15.949   0.992
## s(pc1_last_year)                               29.000  6.061   0.991
## s(log10_gdp_per_capita_constant2010USD_last_year) 29.000  3.382   0.989
##                                                p-value
## s(pc0_last_year)                                  0.32
## s(pc1_last_year)                                  0.27
## s(log10_gdp_per_capita_constant2010USD_last_year)    0.27
```

Transform the target for the GAM that predicts the change in the score on the second principal component

```
cs_pc1_root = gam(abs(change_in_pc1)^root * sign(change_in_pc1) ~ s(pc0_last_year, bs=basis, k=rank) +
```

Again, the quantile-quantile plot now looks much more straight:

```
gam.check(cs_pc1_root)
```



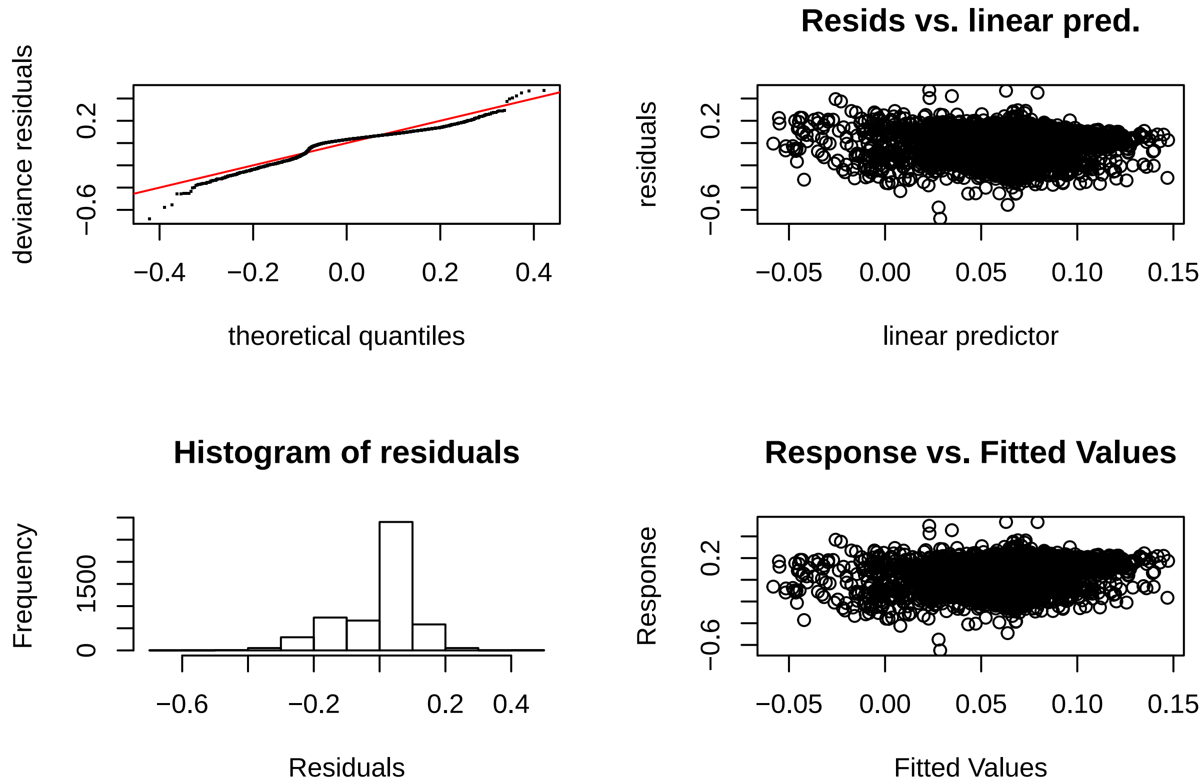

```
##
## Method: GCV   Optimizer: magic
## Smoothing parameter selection converged after 8 iterations.
## The RMS GCV score gradiant at convergence was 4.685651e-08 .
## The Hessian was positive definite.
## The estimated model rank was 88 (maximum possible: 88)
## Model rank =  88 / 88
##
## Basis dimension (k) checking results. Low p-value (k-index<1) may
## indicate that k is too low, especially if edf is close to k'.
##
##                                                      k'   edf k-index
## s(pc0_last_year)                                 29.000 6.481   0.987
## s(pc1_last_year)                                 29.000 7.787   1.025
## s(log10_gdp_per_capita_constant2010USD_last_year) 29.000 1.000   1.011
##                                                  p-value
## s(pc0_last_year)                                    0.18
## s(pc1_last_year)                                    0.98
## s(log10_gdp_per_capita_constant2010USD_last_year)    0.78
```

Transform the target for the GAM that predicts the change in the log-base-10 of per-capita incomes

```
cs_gdp_root = gam(abs(change_in_log10_gdp_per_capita_constant2010USD)^root * sign(change_in_log10_gdp_p
```

Again, the quantile-quantile looks more straight:

```
gam.check(cs_gdp_root)
```




```
##
## Method: GCV   Optimizer: magic
## Smoothing parameter selection converged after 7 iterations.
## The RMS GCV score gradiant at convergence was 4.968686e-07 .
## The Hessian was positive definite.
## The estimated model rank was 88 (maximum possible: 88)
## Model rank =  88 / 88
##
## Basis dimension (k) checking results. Low p-value (k-index<1) may
## indicate that k is too low, especially if edf is close to k'.
##
##                                                  k'    edf  k-index
## s(pc0_last_year)                             29.000 11.099   0.971
## s(pc1_last_year)                             29.000  5.569   0.994
## s(log10_gdp_per_capita_constant2010USD_last_year) 29.000  4.773   1.010
##                                              p-value
## s(pc0_last_year)                                0.00
## s(pc1_last_year)                                0.32
## s(log10_gdp_per_capita_constant2010USD_last_year)    0.78
```

These results suggest that we should transform all three targets with the sign-preserving square root.

Create and save a figure of the QQ plots

Finally, we create and save a figure that illustrates why we chose to apply the square root to the target:

```
make_qq_plot <- function(data, filename, path.to.directory, rank, root = 0.5, title.font.size = 0.9) {
  basis = "cr" # cubic regression spline (CRS)
  cs_pc0 = gam(change_in_pc0 ~ s(pc0_last_year, bs=basis, k=rank) + s(pc1_last_year, bs=basis, k=rank) +
  cs_pc1 = gam(change_in_pc1 ~ s(pc0_last_year, bs=basis, k=rank) + s(pc1_last_year, bs=basis, k=rank) +
  cs_gdp = gam(change_in_log10_gdp_per_capita_constant2010USD ~ s(pc0_last_year, bs=basis, k=rank) + s(p
  cs_pc0_root = gam(abs(change_in_pc0)^root * sign(change_in_pc0) ~ s(pc0_last_year, bs=basis, k=rank) +
  cs_pc1_root = gam(abs(change_in_pc1)^root * sign(change_in_pc1) ~ s(pc0_last_year, bs=basis, k=rank) +
  cs_gdp_root = gam(abs(change_in_log10_gdp_per_capita_constant2010USD)^root * sign(change_in_log10_gdp_

  pdf(paste(path.to.directory, paste(filename, ".pdf", sep = ""), sep = ""), width=6.8, height=4)
  par(mfrow=c(2,3), mai=c(.6, 0.6, 0.45, 0.25)) # bottom, left, top, right
  qq.gam(cs_pc0, main=expression(paste("Predict score on 1st principal component")), cex.main=title.font
  qq.gam(cs_pc1, main=expression(paste("Predict score on 2nd principal component")), cex.main=title.font
  qq.gam(cs_gdp, main=expression(paste("Predict ", log[10], "(GDP per capita)")), cex.main=title.font.si
  qq.gam(cs_pc0_root, main=expression(paste("Predict (score on 1st principal comp.)"^(1/2))), cex.main=t
  qq.gam(cs_pc0_root, main=expression(paste("Predict (score on 2nd principal comp.)"^(1/2))), cex.main=t
  qq.gam(cs_gdp_root, main=expression(paste("Predict (", log[10], "(GDP per capita))"^(1/2))), cex.main=
  dev.off()
}
make_qq_plot(Rpop, "QQ_Rpop", "../../figures/", 30)
```

```
## pdf
##   2
```