FCEF Carbon Analysis

Daniel Marais

2022-08-15

Data Preparation

An overall mass balance, as well as a Carbon-component balance are required to yield a fair estimate for the CO_2 emission quantities in the FeCr-production process.

A simple mass balance layout can be summarised by the following:

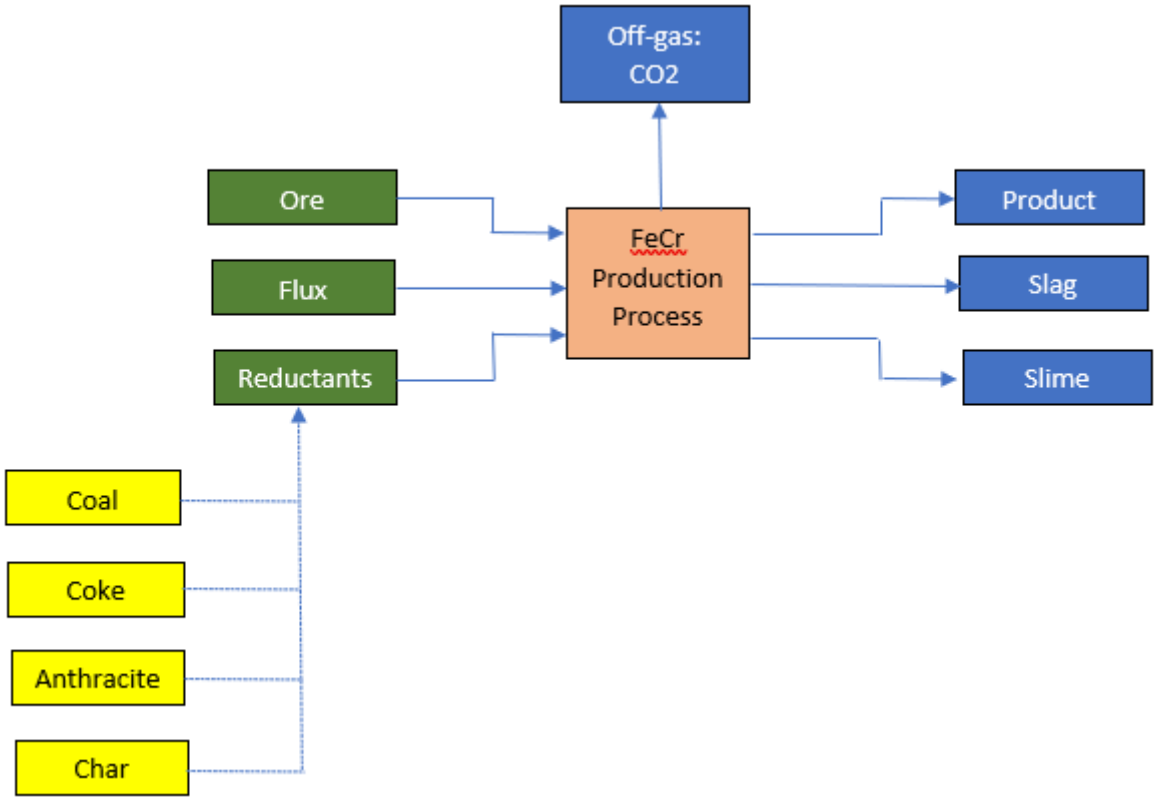


Figure 1: FeCr process mass balance elements

In order to generate a meaningful estimate, the ideal would be to have contemporaneous measurements of all streams and their carbon contents. This is, however, seldom possible as interest in the quantities and qualities of the various input and output streams are not required at the same point in time and off-gas measurements are generally only conducted on special request.

Methodology

A useful way to deal with the fact that contemporaneous data sets cannot be compiled, is to generate distribution statistics of all input and output streams. Based on the assumption that the measured quantities follow a Gaussian distribution, it is only necessary to have means and standard deviations in the recorded statistical data sets for total mass flow and carbon content.

Note that flow quantities of inputs and outputs may be correlated, e.g. a high input of ore, under normal conditions, would be associated with a product stream on the higher end of the distribution scale. Correlations of mass flows are captured by means of a Choleski-transformation of a correlation matrix of mass flows.

Assume no correlation in carbon content of the various input and output mass flows. Formulate stochastic mass balances and carbon component balances based on the assumed gaussian distributed data. Since data is not readily available for off-gas flows and its carbon content, it is always estimated from assumed interaction of all other streams. A stochastic database is generated by considering 100 000 mass and carbon balance simulation cases.

Eliminate outlier cases from the generated database. Also eliminate negative flow and negative carbon content cases.

The aim is to subject the remainder of the generated output (typically ~80% of originally generated simulation cases) to a linear regression analysis to determine the carbon content of the various input and output streams, as provided by the coefficients in the linear model specification. Linear models that include and exclude ore flows are tested – considering the fact that the carbon content in the ore is negligible.
The mean outcomes of the carbon content of inputs and outputs can be adjusted by tweaking the correlation matrix of the various flows.

Finally, one can construct a 90% confidence interval for total CO_2 emissions from the linear regression output.

Procedure with R code

Upload and prepare the data as follows:

```
mass <- read.csv("20220816 Mass flow.csv", header=TRUE)
pctc <- read.csv("20220816 Carbon content.csv", header=TRUE)

# Calculate totals
rownames(mass) <- mass[,1]
mass <- mass[,-1]
mass$Ptm <- apply(mass[,1:5],1,sum)
mass$Ptmsd <- round(apply(mass[,6:10]*mass[,1:5],1,sum)/apply(mass[,1:5],1,sum),0)

rownames(pctc) <- pctc[,1]
pctc <- pctc[,-1]
pctc$Ptc <- apply(pctc[,1:5]*mass[,1:5],1,sum)/apply(mass[,1:5],1,sum)
pctc$Ptcsd <- apply(pctc[,6:10]*mass[,1:5],1,sum)/apply(mass[,1:5],1,sum)

# Design alternative scenarios
#mass$Ptm <- mass$P150m
#mass$Ptmsd <- mass$P150msd
#pctc$Ptc <- pctc$P150c
#pctc$Ptcsd <- pctc$P150csd
```

Assign the uploaded data to the variables which are to be used in the remainder of the algorithm.

```
ore_mass_m <- mass$Ptm[1]
ore_mass_sd <- mass$Ptmsd[1]
ore_pctc_m <- pctc$Ptc[1]
ore_pctc_sd <- pctc$Ptcsd[1]

flx_mass_m <- mass$Ptm[2]
flx_mass_sd <- mass$Ptmsd[2]
flx_pctc_m <- pctc$Ptc[2]
flx_pctc_sd <- pctc$Ptcsd[2]

anth_mass_m <- mass$Ptm[3]
anth_mass_sd <- mass$Ptmsd[3]
anth_pctc_m <- pctc$Ptc[3]
anth_pctc_sd <- pctc$Ptcsd[3]

coal_mass_m <- mass$Ptm[4]
coal_mass_sd <- mass$Ptmsd[4]
coal_pctc_m <- pctc$Ptc[4]
coal_pctc_sd <- pctc$Ptcsd[4]

coke_mass_m <- mass$Ptm[5]
coke_mass_sd <- mass$Ptmsd[5]
coke_pctc_m <- pctc$Ptc[5]
coke_pctc_sd <- pctc$Ptcsd[5]

char_mass_m <- mass$Ptm[6]
char_mass_sd <- mass$Ptmsd[6]
char_pctc_m <- pctc$Ptc[6]
char_pctc_sd <- pctc$Ptcsd[6]

past_mass_m <- mass$Ptm[7]
past_mass_sd <- mass$Ptmsd[7]
past_pctc_m <- pctc$Ptc[7]
past_pctc_sd <- pctc$Ptcsd[7]

slg_mass_m <- mass$Ptm[8]
slg_mass_sd <- mass$Ptmsd[8]
slg_pctc_m <- pctc$Ptc[8]
slg_pctc_sd <- pctc$Ptcsd[8]

slm_mass_m <- mass$Ptm[9]
slm_mass_sd <- mass$Ptmsd[9]
slm_pctc_m <- pctc$Ptc[9]
slm_pctc_sd <- pctc$Ptcsd[9]

prd_mass_m <- mass$Ptm[10]
prd_mass_sd <- mass$Ptmsd[10]
prd_pctc_m <- pctc$Ptc[10]
prd_pctc_sd <- pctc$Ptcsd[10]
```

This data set refers to the 2019 averages obtained for Glencore, i.e. the average over five different sites. The year 2019 was selected because of the relatively high stable production rates recorded, before the disruptive commencement of the Covid-19 pandemic in 2020.

Note that the **mass_m** suffix refers to the monthly mean of the mass flow rates of the input and output streams indicated in the . The **mass_sd** suffix refers to the standard deviation of the monthly mass flow rates. Similarly, the **pctc_m** and **pctc_sd** indicate the means and standard deviations for %-carbon content in the various streams.

Next, set the number of simulations that will be required.

```
n <- 100000
```

The mass flow rate correlation matrix is given by:

```
corMat_mas <- matrix(c(1.0,0.001,0.001,0.001,0.001,0.001,0.001,0.001,0.001,0.001,
                       0.001,1.0,0.0001,0.0001,0.0001,0.0001,0.0001,0.0001,0.0001,0.0001,
                       0.001,0.0001,1.0,0.0001,0.0001,0.0001,0.0001,0.0001,0.0001,0.0001,
                       0.001,0.0001,0.0001,1.0,0.0001,0.0001,0.0001,0.0001,0.0001,0.0001,
                       0.001,0.0001,0.0001,0.0001,1.0,0.0001,0.0001,0.0001,0.0001,0.0001,
                       0.001,0.0001,0.0001,0.0001,0.0001,1.0,0.0001,0.0001,0.0001,0.0001,
                       0.001,0.0001,0.0001,0.0001,0.0001,0.0001,1.0,0.0001,0.0001,0.0001,
                       0.001,0.0001,0.0001,0.0001,0.0001,0.0001,0.0001,1.0,0.0001,0.0001,
                       0.001,0.0001,0.0001,0.0001,0.0001,0.0001,0.0001,0.0001,1.0,0.0001,
                       0.001,0.0001,0.0001,0.0001,0.0001,0.0001,0.0001,0.0001,0.0001,1.0),
                     nrow=10, byrow=TRUE)
```

Note that these correlation settings can be adjusted as a method to tweak the model to deliver the expected flow rates and carbon contents for the various streams.

Estimate the eigenvalues of this correlation matrix:

```
eigen(corMat_mas)$values
```

```
##  [1] 1.0034265 0.9999000 0.9999000 0.9999000 0.9999000 0.9999000 0.9999000
##  [8] 0.9999000 0.9999000 0.9973735
```

Note that all eigenvalues need to be positive for the matrix to be positive definite.

Similarly, define the correlation matrix for the carbon contents of the various streams,

```
corMat_pct <- matrix(c(1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,
                       0.0,1.0,0.00001,0.00001,0.00001,0.00001,0.00001,0.00001,0.00001,0.00001,
                       0.0,0.00001,1.0,0.00001,0.00001,0.00001,0.00001,0.00001,0.00001,0.00001,
                       0.0,0.00001,0.00001,1.0,0.00001,0.00001,0.00001,0.00001,0.00001,0.00001,
                       0.0,0.00001,0.00001,0.00001,1.0,0.00001,0.00001,0.00001,0.00001,0.00001,
                       0.0,0.00001,0.00001,0.00001,0.00001,1.0,0.00001,0.00001,0.00001,0.00001,
                       0.0,0.00001,0.00001,0.00001,0.00001,0.00001,1.0,0.00001,0.00001,0.00001,
                       0.0,0.00001,0.00001,0.00001,0.00001,0.00001,0.00001,1.0,0.00001,0.00001,
                       0.0,0.00001,0.00001,0.00001,0.00001,0.00001,0.00001,0.00001,1.0,0.00001,
                       0.0,0.00001,0.00001,0.00001,0.00001,0.00001,0.00001,0.00001,0.00001,1.0),
                     nrow=10, byrow=TRUE)
```

and run the eigenvalues test.

```
##  [1] 1.00008 1.00000 0.99999 0.99999 0.99999 0.99999 0.99999 0.99999 0.99999
## [10] 0.99999
```

Next, we prepare the Choleski-transformation matrices.

```
choleski_mas = chol(corMat_mas)
choleski_pct = chol(corMat_pct)
```

Now we prepare the matrices that will collect all the simulated random flow rates of the various streams in the mass and carbon balances.

Capture the simulated data in tables for the mass flow rates and carbon contents.

```
tab_mas <- cbind(ore_mas, flx_mas, anth_mas, coal_mas, coke_mas, char_mas, past_mas,
                 slg_mas, slm_mas, prd_mas)
tab_pct <- cbind(ore_pct, flx_pct, anth_pct, coal_pct, coke_pct, char_pct, past_pct,
                 slg_pct, slm_pct, prd_pct)
```

Now apply the Choleski-transformations.

```
tab1_mas <- tab_mas %*% sweep(choleski_mas,2,colSums(choleski_mas),"/")
tab1_pct <- tab_pct %*% sweep(choleski_pct,2,colSums(choleski_pct),"/")
```

The tables need to be cleaned up. First remove extreme values.

```
mahal <- mahalanobis(tab1_mas, colMeans(tab1_mas), na.rm=TRUE, cov(tab1_mas),
                     use="pairwise.complete.obs")
maxcutoff <- qchisq(1-0.001, ncol(tab1_mas))
summary(mahal<maxcutoff)
```

```
##    Mode   FALSE    TRUE
## logical      94   99906
```

```
no.out.tab1_mas <- tab1_mas[mahal < maxcutoff,]
no.out.tab1_pct <- tab1_pct[mahal < maxcutoff,]
```

Next, eliminate rows with negative masses and then also rows with negative carbon contents.

```
if (any(no.out.tab1_mas < 0) == FALSE) {
  pos.no.out.tab1_mas <- no.out.tab1_mas
  pos.no.out.tab1_pct <- no.out.tab1_pct
} else {
  pos.no.out.tab1_mas <- no.out.tab1_mas[-unique(which(no.out.tab1_mas < 0) %%
                                                 nrow(no.out.tab1_mas)),]
  pos.no.out.tab1_pct <- no.out.tab1_pct[-unique(which(no.out.tab1_mas < 0) %%
                                                 nrow(no.out.tab1_mas)),]
}

if (any(no.out.tab1_pct < 0) == FALSE) {
  pos.pos.no.out.tab1_pct <- pos.no.out.tab1_pct
  pos.pos.no.out.tab1_mas <- pos.no.out.tab1_mas
} else {
pos.pos.no.out.tab1_pct <- pos.no.out.tab1_pct[-unique(which(pos.no.out.tab1_pct < 0) %%
                                                       nrow(pos.no.out.tab1_pct)),]
```

```
pos.pos.no.out.tab1_mas <- pos.no.out.tab1_mas[-unique(which(pos.no.out.tab1_pct < 0) %%
                                                        nrow(pos.no.out.tab1_pct)),]
}

tab1_mas <- pos.pos.no.out.tab1_mas; nrow(tab1_mas)/n
```

```
## [1] 0.4718
```

```
tab1_pct <- pos.pos.no.out.tab1_pct; nrow(tab1_pct)/n
```

```
## [1] 0.4718
```

Now estimate the overall mass balance

```
tab1_mas.em <- tab1_mas[,1] + tab1_mas[,2] + tab1_mas[,3] + tab1_mas[,4] + tab1_mas[,5] + tab1_mas[,6] +
  tab1_mas[,7] - tab1_mas[,8] - tab1_mas[,9] - tab1_mas[,10]
tab1_mas.sum <- as.data.frame(cbind(tab1_mas.em, tab1_mas))
colnames(tab1_mas.sum) <- c("em","ore", "flx", "anth", "coal", "coke", "char", "past", "slg", "slm", "p
```

and construct the gas emissions carbon content estimation:

```
tab1_cnt <- as.data.frame(tab1_mas * tab1_pct)
colnames(tab1_cnt) <- c("ore", "flx", "anth", "coal", "coke", "char", "past", "slg", "slm", "prd")
tab1_cnt$em <- tab1_cnt$ore + tab1_cnt$flx + tab1_cnt$anth + tab1_cnt$coal + tab1_cnt$coke +
  tab1_cnt$char + tab1_cnt$past - tab1_cnt$slg - tab1_cnt$slm - tab1_cnt$prd

tab2 <- as.data.frame(cbind(tab1_cnt$em,tab1_mas.sum))
colnames(tab2) <- c("em_cnt", "em_mas", "ore_mas", "flx_mas", "anth_mas", "coal_mas", "coke_mas",
                    "char_mas", "past_mas", "slg_mas", "slm_mas", "prd_mas")
tab2 <- tab2[tab2$em_cnt>=0,]
tab2 <- tab2[tab2$em_mas>=0,]
tab2 <- tab2[tab2$em_cnt<tab2$em_mas,]
```

Determine the number of the remaining records in the table.

```
nrow(tab2)
```

```
## [1] 47179
```

Determine the means and standard deviations of the columns in the table containing carbon flows.

```
apply(tab2,2,mean)
```

```
##     em_cnt     em_mas    ore_mas    flx_mas   anth_mas   coal_mas   coke_mas
##  73459.225 135819.121 346729.978  21593.430  58628.390  13099.532  18862.163
##   char_mas   past_mas    slg_mas    slm_mas    prd_mas
##  10253.259   1734.379 160098.789  24060.193 150923.029
```

```
apply(tab2,2,sd)
```

```
##     em_cnt     em_mas    ore_mas    flx_mas   anth_mas   coal_mas   coke_mas
##  6565.8988 16551.1818 12584.8729  1424.1893  5432.3619  2935.8331  3127.3342
##   char_mas   past_mas    slg_mas    slm_mas    prd_mas
##   809.3912   223.8455  5739.0544  1348.0845  5451.8313
```

Now, estimate the carbon content of the simulated emission gas stream, and its mean and standard deviation:

```
em_pctc <- tab2$em_cnt/tab2$em_mas

mean(em_pctc); sd(em_pctc)
```

```
## [1] 0.5468723
```

```
## [1] 0.06772034
```

The core of the analysis starts with the fitting of a linear model on mass flow rates. The coefficients to the model variables indicate carbon contents.

The first model includes the mass flow rate of the FeCr-ore.

```
fit <- lm(em_cnt ~  0 + ore_mas + flx_mas + anth_mas + coal_mas + coke_mas + char_mas + past_mas +
            slg_mas + slm_mas + prd_mas, data=tab2)
```

Show the results,

```
summary(fit)
```

```
##
## Call:
## lm(formula = em_cnt ~ 0 + ore_mas + flx_mas + anth_mas + coal_mas +
##     coke_mas + char_mas + past_mas + slg_mas + slm_mas + prd_mas,
##     data = tab2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -14638.6  -2123.9     -1.6   2126.0  12457.2
##
## Coefficients:
##            Estimate Std. Error t value Pr(>|t|)
## ore_mas   0.0004868  0.0010313   0.472   0.6369
## flx_mas   0.1095201  0.0099335  11.025  < 2e-16 ***
## anth_mas  0.8315900  0.0026526 313.502  < 2e-16 ***
## coal_mas  0.7302622  0.0049907 146.325  < 2e-16 ***
## coke_mas  0.8578399  0.0046704 183.677  < 2e-16 ***
## char_mas  0.7981114  0.0177390  44.992  < 2e-16 ***
## past_mas  0.7886819  0.0654251  12.055  < 2e-16 ***
## slg_mas  -0.0040637  0.0022243  -1.827   0.0677 .
## slm_mas  -0.0807599  0.0103526  -7.801 6.27e-15 ***
## prd_mas  -0.0697932  0.0023478 -29.727  < 2e-16 ***
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3191 on 47169 degrees of freedom
## Multiple R-squared:  0.9981, Adjusted R-squared:  0.9981
## F-statistic: 2.516e+06 on 10 and 47169 DF,  p-value: < 2.2e-16
```

and determine the 90% confidence intervals on the coefficients (carbon contents).

```
confint(fit, level=0.9)
```

```
##                   5 %          95 %
## ore_mas  -0.001209449  0.0021831331
## flx_mas   0.093180634  0.1258596566
## anth_mas  0.827226801  0.8359531872
## coal_mas  0.722053131  0.7384713636
## coke_mas  0.850157650  0.8655221217
## char_mas  0.768932690  0.8272900420
## past_mas  0.681065000  0.8962987420
## slg_mas  -0.007722455 -0.0004049468
## slm_mas  -0.097788692 -0.0637311563
## prd_mas  -0.073655006 -0.0659313777
```

Repeat the model fitting, now excluding the ore mass flow:

```
fits <- lm(em_cnt ~  0 + flx_mas + anth_mas + coal_mas + coke_mas + char_mas + past_mas +
             slg_mas + slm_mas + prd_mas, data=tab2)
```

Show the results of this model.

```
summary(fits)
```

```
##
## Call:
## lm(formula = em_cnt ~ 0 + flx_mas + anth_mas + coal_mas + coke_mas +
##     char_mas + past_mas + slg_mas + slm_mas + prd_mas, data = tab2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -14648.2  -2123.6     -0.6   2127.5  12445.3
##
## Coefficients:
##           Estimate Std. Error t value Pr(>|t|)
## flx_mas   0.110262   0.009808  11.242  < 2e-16 ***
## anth_mas  0.831720   0.002638 315.252  < 2e-16 ***
## coal_mas  0.730357   0.004987 146.465  < 2e-16 ***
## coke_mas  0.857960   0.004663 183.976  < 2e-16 ***
## char_mas  0.799282   0.017565  45.505  < 2e-16 ***
## past_mas  0.792528   0.064915  12.209  < 2e-16 ***
## slg_mas  -0.003748   0.002121  -1.767   0.0773 .
## slm_mas  -0.079827   0.010162  -7.855 4.07e-15 ***
## prd_mas  -0.069462   0.002241 -31.000  < 2e-16 ***
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3191 on 47170 degrees of freedom
## Multiple R-squared:  0.9981, Adjusted R-squared:  0.9981
## F-statistic: 2.795e+06 on 9 and 47170 DF,  p-value: < 2.2e-16
```

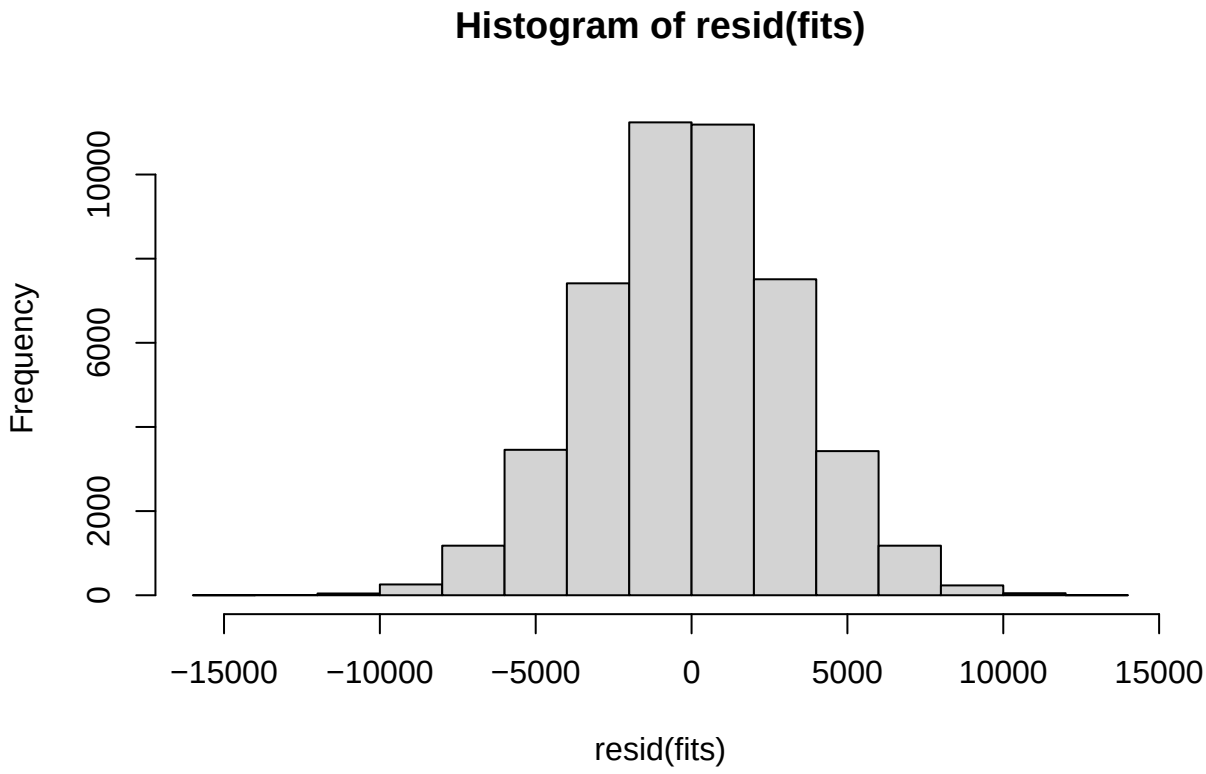
Now, determine the 90% confidence intervals on the coefficients (carbon contents).

```
print(confint(fits, level=0.9), digits=4)
```

```
##                5 %       95 %
## flx_mas   0.094129  0.1263957
## anth_mas  0.827380  0.8360594
## coal_mas  0.722155  0.7385598
## coke_mas  0.850289  0.8656306
## char_mas  0.770390  0.8281739
## past_mas  0.685750  0.8993065
## slg_mas  -0.007237 -0.0002586
## slm_mas  -0.096543 -0.0631116
## prd_mas  -0.073148 -0.0657766
```

Check the histogram of the residual values from this model.

```
hist(resid(fits))
```



Estimate the 90% confidence interval for CO_2 emissions from the process. The estimate reflects the total emissions from the five Glencore sites.

```
em_cnt_mean <- apply(tab2,2,mean)[1]
em_cnt_sd <- apply(tab2,2,sd)[1]
em_res <- c(em_cnt_mean-1.96*em_cnt_sd, em_cnt_mean, em_cnt_mean+1.96*em_cnt_sd)*
  44/12*12
names(em_res) <- c("5%", "mean", "95%")
print(em_res, digits=5)
```

```
##      5%    mean     95%
## 2665963 3232206 3798449
```

Next we load fuel combustion emission factors - units: ton CO_2 per Mton of FeCr produced.

Provide log-means and log-standard deviations for the various fuel components.

```
Propane_mean <- 5.333
Propane_sd <- 0.406
LFO_mean <- 7.304
LFO_sd <- 0.169
HFO_mean <- 8.915
HFO_sd <- 0.101
Kero_mean <- 3.319
Kero_sd <- 0.975
LPG_mean <- 4.972
LPG_sd <- 2.798
Coal_mean <- 11.394
Coal_sd <- 0.159
Diesel_mean <- 9.192
Diesel_sd <- 0.046
Petrol_mean <- 3.023
Petrol_sd <- 0.304
```

State the number of simulations required.

```
m <- nrow(tab2)
```

Next, determine the annual production of CO_2 from fuel combustion

```
propCO2 <- exp(rnorm(m,Propane_mean,Propane_sd))*tab2$prd_mas
lfoCO2 <- exp(rnorm(m,LFO_mean,LFO_sd))*tab2$prd_mas
hfoCO2 <- exp(rnorm(m,HFO_mean,HFO_sd))*tab2$prd_mas
keroCO2 <- exp(rnorm(m,Kero_mean,Kero_sd))*tab2$prd_mas
lpgCO2 <- exp(rnorm(m,LPG_mean,LPG_sd))*tab2$prd_mas
coalCO2 <- exp(rnorm(m,Coal_mean,Coal_sd))*tab2$prd_mas
dieselCO2 <- exp(rnorm(m,Diesel_mean,Diesel_sd))*tab2$prd_mas
petrolCO2 <- exp(rnorm(m,Petrol_mean,Petrol_sd))*tab2$prd_mas

combustCO2 <- (propCO2+lfoCO2+hfoCO2+keroCO2+lpgCO2+coalCO2+dieselCO2+petrolCO2)*12/10^6
```

Eliminate outliers from combustion CO_2 distribution and plot the cleaned up data profile.

```
Q3 <- quantile(combustCO2,0.75)
Q1 <- quantile(combustCO2,0.25)
iqr <- IQR(combustCO2)
combustCO2.clean <- subset(combustCO2,combustCO2>(Q1-1.75*iqr) & combustCO2<(Q3+1.75*iqr))

plot(density(combustCO2.clean), main="")
```

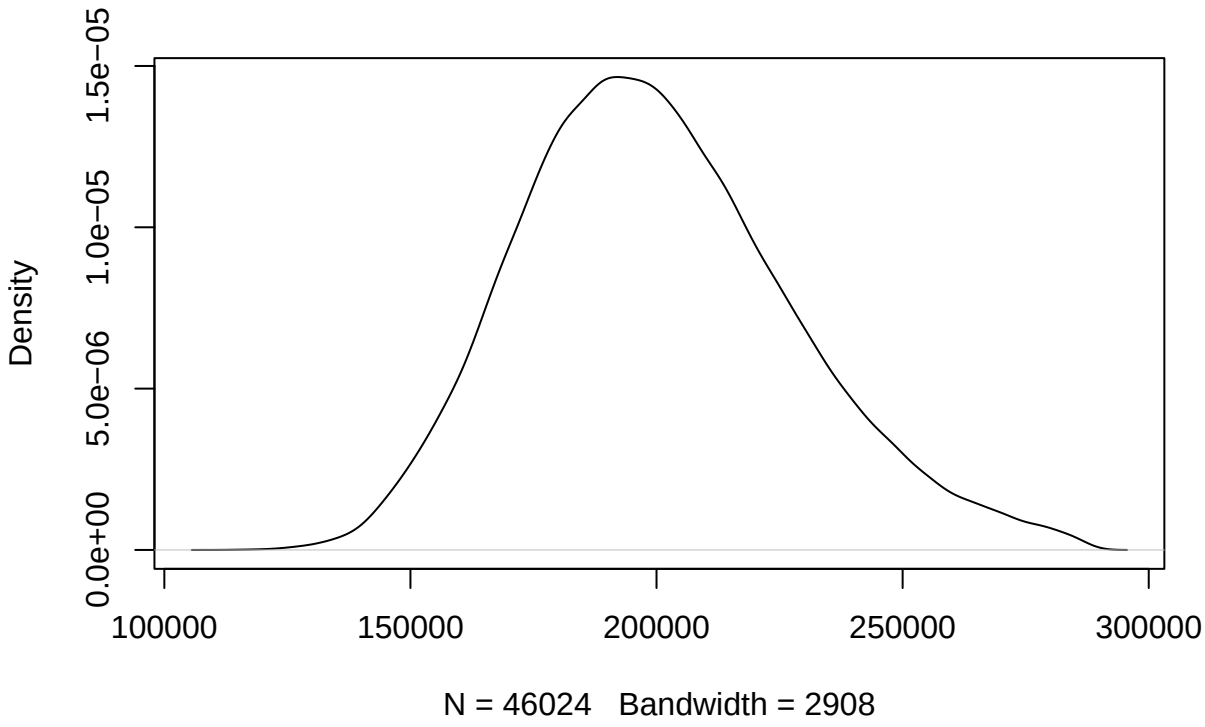


Figure 2: Cleaned up CO_2 from fuel combustion

Estimate total CO_2 production range and plot the density profile.

```
totCO2 <- (tab2$em_cnt[1:length(combustCO2.clean)]*44/12*12+combustCO2.clean)
plot(density(totCO2), main="")
```

Finally, estimate the 90% confidence interval for CO_2 emissions from the process and fuel combustion. The estimate reflects the total emissions from the five Glencore sites.

```
totCO2_mean <- mean(totCO2)
totCO2_sd <- sd(totCO2)
totCO2_res <- c(totCO2_mean-1.96*totCO2_sd, totCO2_mean, totCO2_mean+1.96*totCO2_sd)
names(totCO2_res) <- c("5%", "mean", "95%")
print(totCO2_res, digits=5)
```

```
##      5%    mean     95%
## 2863082 3432016 4000950
```

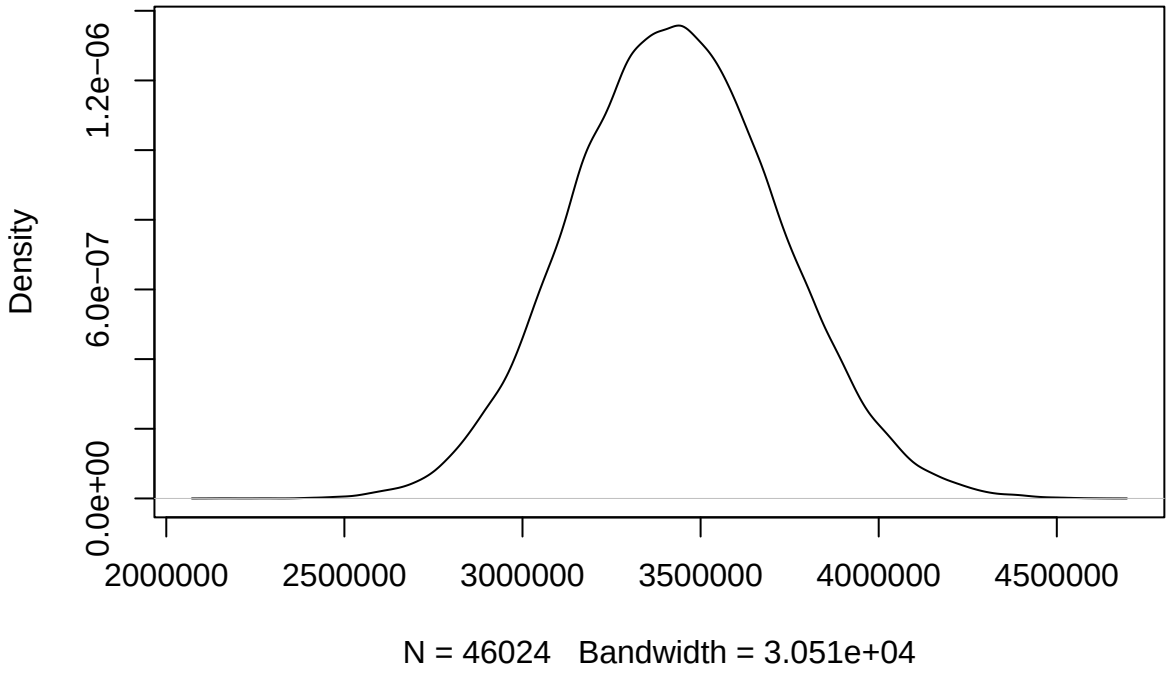


Figure 3: Profile of total CO_2 emission

Table 1: Comparison of estimated and observed CO 2 emissions

	FeCr process		Total emissions	
	Estimated	Observed	Estimated	Observed
5%	2665963	0	2863082	0
mean	3232206	3252225	3432016	3631164
95%	3798449	0	4000950	0

Table 2: Comparison of estimated and observed carbon contents in streams

	FeCr process	
	Estimated	Observed
ore_mas	0.0004868	0.0000100
flx_mas	0.1095201	0.1156761
anth_mas	0.8315900	0.8279532
coal_mas	0.7302622	0.7265299
coke_mas	0.8578399	0.8675041
char_mas	0.7981114	0.7762846
past_mas	0.7886819	0.8671320
slg_mas	0.0040637	0.0021510
slm_mas	0.0807599	0.0830814
prd_mas	0.0697932	0.0687865

Show summary of results and compare to actual observations (Table 1).

```
finalres <- data.frame(em_res,c(0,3252225,0),totCO2_res,c(0,3631164,0))
colnames(finalres) <- c("Estimated","Observed","Estimated","Observed")

#knitr::kable(finalres, "pipe", col.names=c("Est FeCr","Obs FeCr", "Est Total","Obs Total"),
#             caption="Comparison of estimated and observed CO~2~ emissions")

kbl(finalres, align=c("c","c","c","c"), caption="Comparison of estimated and
    observed CO~2~ emissions") %>%
  kable_paper(full_width = F) %>%
  add_header_above(c(" "= 1, "FeCr process" = 2, "Total emissions" =2))
```

Generate comparisons of estimated and observed carbon contents (Table 2).

```
cc_res <- data.frame(abs(coef(fit)),c(ore_pctc_m, flx_pctc_m, anth_pctc_m, coal_pctc_m, coke_pctc_m,
                                      char_pctc_m, past_pctc_m, slg_pctc_m, slm_pctc_m, prd_pctc_m))

kbl(cc_res, col.names=c("Estimated","Observed"),
             caption="Comparison of estimated and observed carbon contents in streams") %>%
  kable_paper(full_width=F) %>%
  add_header_above(c(" " = 1, "FeCr process" = 2))
```

Concluding remarks

Mass and carbon balances can be constructed for aggregate cases, as well as cases for individual production plants. It will, however, require assumptions for variable distributions in cases where statistical measured

data does not exist – it is proposed to assume aggregate averages in such instances, as constructed for the Glencore aggregate case.