# Manual
# **PKP**

Martin Pollack, Michele Vascellari

December 13, 2012

# Contents

1 User Manual 3
1.1 First Steps . . . 3
1.2 Generate Results . . . 4
1.2.1 Using the Graphical User Interface . . . 5
1.2.2 Using the input files . . . 8
1.2.3 The Result Files . . . 11

2 Equations 12
2.1 Pyrolysis Kinetic Equations . . . 12
2.1.1 The Constant Rate Model . . . 12
2.1.2 The Arrhenius Model . . . 13
2.1.3 The Kobayashi Model . . . 14
2.1.4 The Distributed Activation Energy Model . . . 16
2.1.5 Fitting the Kinetic Equations . . . 18
2.1.6 Pyrolysis Species- and Energy Conservation for CPD output . . . 19
2.1.7 Pyrolysis Species and Energy Conservation for FG-DVC output . . 20

3 Programs Structure 22

# 1 User Manual

PKP, the Pyrolysis Kinetic Preprocessor, is a tool to generate the parameter for modeling the pyrolysis process in CFD-applications. It allows a central data input for all three connected pyrolysis programs CPD, CPD and PC Coal Lab. The results of the selected pyrolysis programs will be used to generate the kinetic parameter for several kinetic models: constant Rate, Arrhenius rate, Kobayashi model and the Distributed Activation Energy Model (DAEM), see chapter 2.1. A second procedure generates parameter for the species and energy balance, chapters 2.1.6 and 2.1.7.
PKP is written object orientated in Python 2.7.3.

## 1.1 First Steps

Before running the CPD pyrolysis model with the program the first time, the CPD -Code[1] has to be compiled and the executable has to copied to the main directory (containing also the Python code). There the CPD executable should be renamed *CPD.out*.[2] The directory must contain the following files:

- Coal.inp (User input for the information about the coal)
- CPD.inp (User input for the settings of CPD )
- CPD.out (the CPD executable)
- FGDVC.inp (User input for the settings of FG-DVC )
- IN.dat (input file for CPD)
- OperCond.inp (User input for the operating conditions)
- Pyrolysis.py (the main code, accessing to the classes defined in the other *.py)

The subdirectory *src* includes the following Python files, *Pyrolysis.py* accesses to:

- Compos_and_Energy.py
- CPD_Result.py
- CPD_SetAndLaunch.py
- FGDVC_Result.py
- FGDVC_SetAndLaunch.py
- FitInfo.py
- Fitter.py
- InformationFiles.py

[1] available at: http://www.et.byu.edu/~tom/cpd/cpdcpnlg/cpdcp_nlgfiles.html

[2] Alternatively the name of the executable can be changed in the *Pyrolysis.py*. But rename the file is the less effort.

- Models.py

Before running FG-DVC, the directories have to be defined, see the paragraph **FGDVC.inp** in chapter 1.2. There the two information for **main directory FG-DVC:** and **print increment, writeValue:** must be defined. For CPD **print increment, writeValue:** must be defined. All these information are kept, also when using the GUI as input. So they just have to be defined once as long as they don't need to be changed.

## 1.2 Generate Results

The output of the pyrolysis programs has to be individually modified for a further work with. Also the user of PKP should be aware of the following points:

**FG-DVC** PKP was tested using `FG-DVC` version 8.2.3. Also the output of `FG-DVC` 8.2.2., which seems to be concerning the shape identical to 8.2.3, was tested.

- `FG-DVC` merge the olefins and paraffins into the rate of tar. But the yields of tar, outputted by `FG-DVC` just consider the species of tar except the olefins and paraffins. As the olefins and paraffins are in general not considered as single species in CFD combustion and gasification applications, the yield of tar generated by `FG-DVC` further also includes the Olefins and Paraffins. As the output of the olefins and paraffins themselves are correct, the kinetic parameter of this two species are additionally still fitted.
- The output of `FG-DVC` does not directly report the yields and rates of hydrogen. To generate this information, the yields of hydrogen are calculated by subtract the light gases ($CH_4$, CO, $CO_2$, $H_2O$, ...) from the total yields (last column in the 'gasyield.txt'). The rate is gained by derive a smoothed curve[3].
- The amount of moisture and ash is both equal zero. This is recommended in the manual. Trying to set a fraction of moisture and ash leads to unrealistic results in the `FG-DVC` output (negative yields and rates).
- The constant numerical time step for `FG-DVC` defined by the user shouldn't be lower than $10^{-3}$. The temperature history defined by the user is imported by `FG-DVC` as a text file[4]. According to experience can `FG-DVC` not read imported text files defining the temperature history with a precision better than the $10^{-3}$.

**CPD**

- The outputted result files of `CPD` does contain various information. For the further work the fraction of yields (beginning with 'f', e.g. 'ftar' in the first and fourth `CPD` output file are further observed.)

[3] The subtraction of the other species leads to small oscillation in the yields. These yield curves are further usable. But deriving this curve strengthen this oscillation to a giant value. So the yield curve has to be smoothed before deriving it by apply the equation $y_i = \frac{1}{2}\alpha \cdot (y_{i-1} + y_{i+1}) + (1-\alpha) \cdot y_i$ fifty times over all points of the yields. This has still a low influence on the results (applying it very often would lead to a flat curve), but decreases the oscillation that much, that they are small at the derived curve (central differencing scheme).

[4] 'tTHistory.txt' which was generated by PKP and written into the `FG-DVC` main directory.

- As the `CPD` output does not contain the rates, these rates over time are generated by deriving them using the CDS.
- If an oscillation in the output of the rates occur is the time step too low. There are not enough counting numbers in the output file. So the reported rates vary from zero (the in the CDS subtracted two yields have the same value) and a very high value (the last counting number changes, but the time step is very low).
  The user has here the following possibilities to solve this problem:
  - Use a lower time step.
  - Set '*print increment, writeValue:*' in *CPD.inp* to a high value. The calculation is still carried out with the ow time step. But not every calculated time is written into the result file.
  - Modify the precision of the `CPD` output by modify the `CPD` code. If just the number of counting numbers is modified PKP will still be able to read this modified output.

The user of PKP can apply a number of runs up to five of the pyrolysis programs. Every runs must have its individual temperature history. The different devolatilization results caused by the different operating conditions shall ensure a better applicability of the kinetics in CFD simulations, where the operating conditions can vary over a range. Taking responsibility to the computational costs, two or three runs is a good option to generate the kinetic parameter.

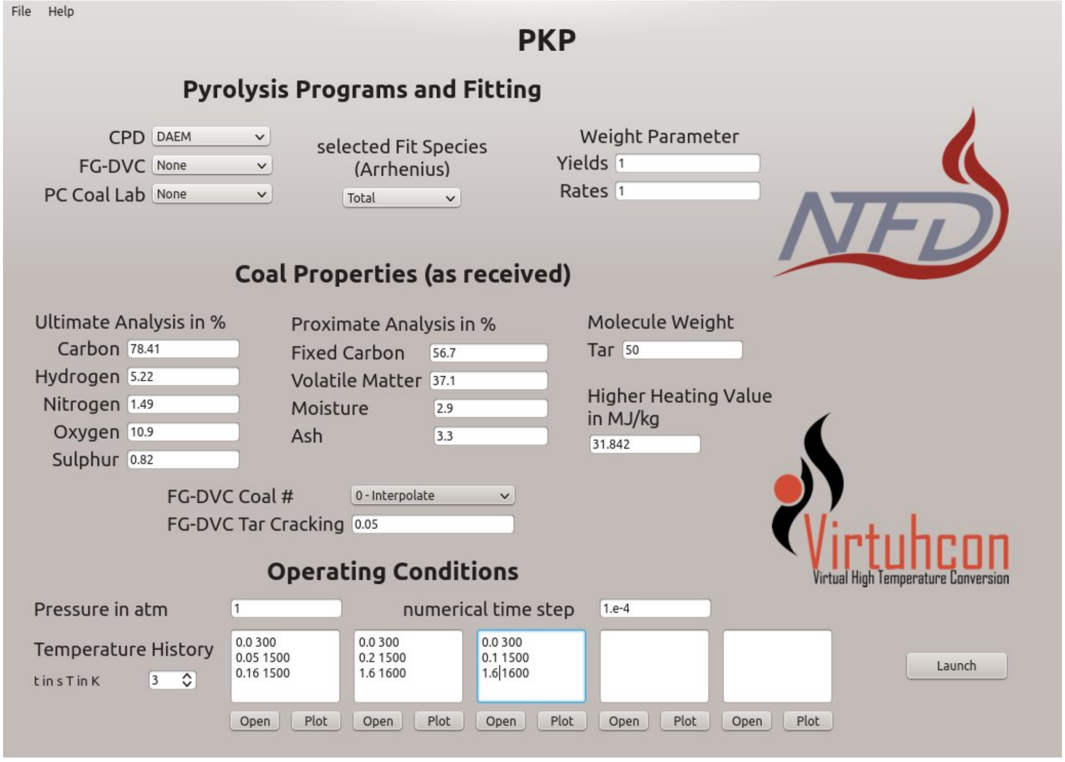
The directly outputted results of the pyrolysis programs are copied into the main PKP directory. The number of the run is contained in the filename.

The following subsections give mode detailed information and an overview on the specific user input and the generated PKP results.

### 1.2.1 Using the Graphical User Interface

When starting the graphical user interface, file *PKPgui.py*, the central input windows opens, see figure 1. As also visible in this figure, the general input structure is splitted into three parts: *Pyrolysis Programs and Fitting*, *Coal Properties* and *Operating Conditions*. The following selections can be done in the first of these:

- **CPD**, **FG-DVC**, **PC Coal Lab**: These column boxes allow a selection, which of these detailed models shall be used and which kinetic parameter shall be fitted. The first two possible selections are here *None*, for not running this model, *Run* to just run it and make the species analysis. A fitting of kinetic parameter will not be carried out. Additional selections for fitting the results are *constant Rate* (eq. 2), *Arrhenius* (eq. 7), *Arrhenius no B* (eq. 9), *Kobayashi* (eq. 10) and *DAEM* (eq. 12). Here the fitting and the species analysis will be done.
- **selected Fit Species (Arrhenius)** is only important if one of the selected models to fit is *Arrhenius* or *Arrhenius no B*. This column box allows which species shall be fitted. Possible options to choose are here:
  - *Total*: only the overall yield is fitted



PKP

Pyrolysis Programs and Fitting

CPD: DAEM
FG-DVC: None
PC Coal Lab: None

selected Fit Species (Arrhenius): Total

Weight Parameter
Yields: 1
Rates: 1

Coal Properties (as received)

Ultimate Analysis in %
Carbon: 78.41
Hydrogen: 5.22
Nitrogen: 1.49
Oxygen: 10.9
Sulphur: 0.82

Proximate Analysis in %
Fixed Carbon: 56.7
Volatile Matter: 37.1
Moisture: 2.9
Ash: 3.3

Molecule Weight
Tar: 50

Higher Heating Value in MJ/kg: 31.842

FG-DVC Coal #: 0 - Interpolate
FG-DVC Tar Cracking: 0.05

Operating Conditions

Pressure in atm: 1
numerical time step: 1.e-4

Temperature History
t in s T in K: 3

```
0.0 300     0.0 300     0.0 300
0.05 1500   0.2 1500    0.1 1500
0.16 1500   1.6 1600    1.6|1600
```

Figure 1: A screenshot of the Input GUI.

  - *Main Species*: the yield curves of the overall yields, tar and gas (the sum of the light gases) are fitted
  - *all Species*: the kinetic parameter for all species outputted by the detailed model are fitted

- **Weight Parameter** Enter the parameter for the fitting procedure. The text line Weight Parameter Yields sets the factor $a_0$ in equation 15, Rates sets $a_1$ in equation 16.

The second section of the GUI, the information about the coal can be entered. All properties refer to the coal in the as received state.

- **Ultimate Analysis in %** Sets the ultimate analysis of the coal. Values should be in percent.

- **Proximate Analysis in %** Sets the proximate analysis of the coal. Values should be in percent.

- **Molecule Weight Tar**: define the molecule weight of the tar. Required for the species analysis.

- **Higher Veating Value** Enter here the higher heating value of the coal in $\frac{\mathrm{MJ}}{\mathrm{kg}}$.

- **FG-DVC Coal #** FG-DVC needs an information whether the coal property file should be interpolated using the van-Krevelen Diagram (option *0-Interpolate*) or if a library coal should be used (1-8). If the selected coal is out of the range of the library coals, the closest library coal has to be selected. The generation of the coal file is also done automatically by PKP.[5]

- **FG-DVC tar cracking** Defines the FG-DVC modeling of the tar cracking. Possible selections are:
  - *0.0* No tar cracking is modeled.[6]
  - When entering a time in seconds, e.g. '0.01', defines the holding time of the tar in the coal molecule. The tar is cracked during this time frame.
  - A negative number, e.g. '-1', sets the full tar cracking.

The last section in the GUI allows the input of the operating conditions:

- **Pressure in atm** Defines the pressure where the devolatilization occurs.
- **numerical time step** Sets the time step. For FG-DVC the constant value, for CPD the maximum.
- **Temperature History** With the spin box on the left, the numbers of temperature histories to apply is defined. If this is set to '3', three runs are carried out using the first three temperature histories. These temperature histories can be imported via the five text fields on the right. There the temperature histories can be entered manually or also be imported using the 'open' buttons in below. Additionally they can be plotted (button 'plot'). Using this option also saves the temperature history in a file *TempHist1.dat*.Where the number stands for the current field. These saved temperature histories can be re-imported using the 'open' button.
  All the temperature histories have to be entered in two columns, just separated by spaces. The time is in seconds, the temperature in Kelvin.

The head menu contains under File following options:

| | |
|---|---|
| Write into Table | Ctrl+S |
| Write and Run | Ctrl+R |
| Load saved state | Ctrl+O |
| Show generated Results | Ctrl+T |
| Exit | Ctrl+Q |

Where the Write saved all information into the *.inp* files and the *TempHist#.dat* in the main directory. 'Load saved state' transfer this information into the current GUI.
The 'Write and Run' is identical with the button 'Launch' in the lower right of the GUI.
The file menu 'Help' offers to open this manual.

Please do not forget to make the general settings in the input files, before using PKP the first time, see chapter 1.1.

After running the result windows opens, figure 2. It lists all calculated species in the column bar. With 'Show results' the yields over time for the detailed model output and

[5] Using the FG-DVC *coalsd.exe*. For more information see the FG-DVC manual.

[6] The option recommended by the FG-DVC manual [1].

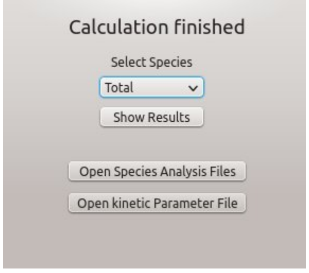


Figure 2: A screenshot of the Result GUI.

the fitted equation are plotted. 'Open Species analysis' opens the textiles with the species analysis results for each run. Open kinetic parameter opens the text file with the results of the fitting, one file for each detailed model. If this window was closed, it can be reopened by the option 'Show generated Results' (Ctrl+T) in the Main window.
All results are located in the Result directory. When starting a new run, its content is deleted.

### 1.2.2 Using the input files

The manual user input is managed by the four **.inp* files, *Coal.inp, CPD.inp FGDVC.inp* and *OperCond.inp*.
The information you want to insert into these files have to be in the line below the line asking for the information. For example:

*Fixed Carbon:*
*43.7*

This sets the amount of Fixed Carbon equal 43.7. The position (i.e. the line) of such two lines in the input file does not matter, the only important point is the specific string (in this example *Fixed Carbon:*) and that the value you want to set is in the following line after the string.[7]
Firstly you will get a short overview into these files and the values to enter into them:

**Coal.inp** contains the main information about the coal. PKP forwards the information about the coal from this file to the programs `CPD` and the `FG-DVC` coal generator (*/COALS/coalsd.exe* in the `FG-DVC` directory). The proximate analysis values are only required by `CPD`.

- **Fixed Carbon:** sets the amount of fixed carbon in the coal. The value has to be entered in percent for a non-daf coal.
- **Volatile Matter:** sets the amount of volatile matter in the coal in percent, as received.

[7] If you want to use another string in this file, you also have to change the individual file note in *InformationFiles.py*.

- **UA Carbon:**, **UA Hydrogen:**, **UA Nitrogen:**, **UA Oxygen:** sets the ultimate analysis for the coal to model. The values have to be entered in percent.
- **Higher Heating Value, as received, in J/kg:** sets the higher heating value for the coal. If this value is not known, set it equal zero. Then the Dulong formula (equation 23) will be used to calculate the higher heating value.
- **Tar Molecule weight, MTar:** Sets the molecule weight of the tar, as it is required for the species and energy calculation, see chapters 2.1.6 and 2.1.7.
- **Weight-Parameter yields for fitting the kinetics:** sets the weight $a_0$ of the equation 15 to weight the yields in the fitting procedure (equation 14).
- **Weight-Parameter rates for fitting the kinetics:** sets the weight $a_1$ of the equation 16 to weight the rates in the fitting procedure (equation 14).

**CPD.inp** controls the `CPD` program and the further work with its output.

- **useCPD?:** if set to *yes* or *true*, `CPD` will be launched.
- **selected fitting Approximation:** if the *constantRate* is selected, the fitting will be carried out using equation 3 and 4. When selecting *Arrhenius*, the kinetic parameter for the Arrhenius equation modeled pyrolysis kinetics (equation 7) will be calculated. To fit the Kobayashi parameter, set it to *Kobayashi*. If selecting *None*, no fitting of the kinetic parameter will be carried out, just the direct `CPD` results and species and energy balance will be generated.
- **initial time step in s:** The initial time step, `CPD` starts to calculate with.
- **print increment, writeValue:** Integer which sets the frequency of writing the result into the `CPD`-output file. E.g. '1' means every value is written into the file, '3' only every third value.

**FGDVC.inp** controls the `FG-DVC` program and the further work with its output.

- **use FG-DVC?:** if set to *yes* or *true*, `FG-DVC` will be launched.
- **selected fitting Approximation:** This selection is analogous to the in *CPD.inp*
- **main directory FG-DVC:** sets the main path of `FG-DVC`, where the *fgdvc.exe* is located. One example: 'C:\Programs\FGDVC_8-2-3\'
- **directory fgdvc-output:** Sets the main directory, where `FG-DVC` outputs the results. This is in general the directory, where the *fgdvcd.exe* is located. One example: 'C:\Programs\FGDVC_8-2-3\FGDVC\'.
  To use other already generated FG-DVC output files, it is a good option to set their path here. As long as they are still named *gasrate.txt* and *gasyield.txt*, the fitting will be carried out on the information contained in these files.

- **Choose Coal:** If this is set equal *0*, the interpolation of the coal will be carried out using the information from *Coal.inp* and the FG-DVC program *coalsd.exe*, leading to specific `FG-DVC` coal files for the applied coal. If the file cannot be generated, i.e. the used coal is outside the interpolation triangle[8], select a value from *1* to *8* to use one of the `FG-DVC` library coals. The order here is the same as in `FG-DVC`:

  1. Beulah-Zap
  2. Wyodak-Anderson
  3. Illinois # 6
  4. Bind Canyon, UT
  5. Lewis-Stockton, WV
  6. Pittsburgh # 8
  7. Upper Freeport, PA
  8. Pocahontas # 3, VA

- **Model tar cracking?** To model no tar cracking (as recommended in the `FG-DVC` manual) set the tar residence time equal *0*. A partial tar cracking can be modeled by set the tar residence time is seconds. If a full tar cracking shall be used, set the residence time to a negative input value, e.g write *-1*.

**OperCond.inp** sets the operating condition for the pyrolysis programs.

- **pressure in atm:** Sets the constant pressure in atmospheres.
- **FG-DVC: constant (numerical) time step; CPD: maximum time step**: Enter here the numerical time step, the constant for `FG-DVC` and the maximum value for `CPD`.
- **Number of Temperature Histories to include:** Defines, how many runs of the pyrolysis models should be carried out. The different temperature history will be defined with the help of the next point:
- **Start Time History 1:** This line have to follow two columns, defining the temperature history for the first run of the pyrolysis models. The first column lists the time in seconds, the second one the temperature in K. The last time point is automatically selected as the final pyrolysis time. The end of the time-temperature array has to be labeled by the term *End Time History 1*. Here one example:
  *Start Time History 1*
  *0, 293*
  *0.05, 1000*
  *0.1, 1700*
  *0.6, 1700*
  *End Time History 1*
  Analogue to this term the numbers 2 to 5 label the temperature history for the second to the fifth run.

[8] see the FG-DVC manual for more details: THE FG-DVC COAL PYROLYSIS MODEL USER'S GUIDE Version 8.2.3 for Windows; Advanced Fuel Research, Inc., 87 Church Street, East Hartford, CT 06108-3728, USA; 2012

### 1.2.3 The Result Files

The generated results are documented in the following files. The name of the text-files contains in front the used pyrolysis program (e.g. *CPD-BalanceResults.txt*).

**BalanceResults.txt** This file contains the output of the species and the energy conservation, chapters 2.1.6 and 2.1.7. The first part lists the input of the UA, PA and HHV. Afterwards, the final yields of all species are enumerated, as the result of using the equation 20 to 21. The tar composition (equation 22), assuming a $C_nH_mO_p$ molecule with an average molecule mass, inputted by the user, is given by the factors n, m, p. The last two parts show the results applying the equations 23 to 32.

**Results_const_rate.txt** This file contains the kinetic parameters for the constant rate (equation 3 or 4) fitting. The two parameters are k in $\frac{1}{s}$ and $t_{start}$ in s.

**Result_ArrheniusRate.txt** *Result_ArrheniusRate.txt* lists all kinetic parameter (A in $\frac{1}{s}$ and E in K) for the Arrhenius equation 6 or 7.

**Results_KobayashiRate.txt** This file list for all species the Kobayashi kinetic parameter $A_1$ and $A_2$ in $\frac{1}{s}$, $E_1$ and $E_2$ in K, $\alpha_1$ and $\alpha_2$.

**Fit_result_[Species]_[R/Y].pdf** These files contains the plots of the pyrolysis program output curve and the estimated curve using the applied model. These plots show the rate (then *_R.pdf) or yields (*_Y.pdf). This plot exists for all species calculated by the pyrolysis program (named in *_Species_*).

**Fit_result_[Species].out** For every species is in the referring file the *Time*, *Temperature*, *Yields* and the *Rates* written in columns. This is not the original output from the pyrolysis program, it is the result applying the selected equation with the fitted parameter.

Table 1: The fitted models and their parameter.

| Model | fitted parmeter | fitted species |
|---|---|---|
| constant Rate | $k_i$, $t_{start,i}$, $m_{final}$ | all |
| Arrhenius | $A_i$, $b_i$, $E_i$, $m_{final}$ | all |
| Kobyashi | $A_1, E_1, A_2, E_2$ | total |
| DAEM | | |

# 2 Equations

## 2.1 Pyrolysis Kinetic Equations

In PKP are four possible kinetic models to select:

- The Constant Rate Model, section 2.1.1
- The Arrhenius Model, 2.1.2
- The Kobayashi Model, 2.1.3
- The Distributed Activation Energy Model, 2.1.4

The Kobayashi model has the advantage, that the yields are dependent on the individual temperature history. A change, may in in the final temperature leads to a higher yield.
The constant rate and the Arrhenius model has fixed final yields. The general devolatilization reactions for these two models are described with equations 1 and 2, where 1 describes the mass loss of the solid (index $s$) coal, equation 2 the formation of the yields (gaseous:$g$, individual:$i$).

$$\frac{dm_s}{dt} = -k_s\left(m_s - m_{s,final}\right) \tag{1}$$

$$\frac{dm_{g,i}}{dt} = k_{g,i}\left(m_{g,i,final} - m_{g,i}\right) \tag{2}$$

So when using more than one run, also the final yield is a parameter to fit. It will be located in between the final yields of the runs of the pyrolysis models.

In the following subsections, the plots showing the rates and yields were generated with `CPD` and `FG-DVC` showing the yields and rates for different species. All are based on the same temperature profiles as shown in figure 3.

The table 2.1 gives an overview of the models with their parameters to fit and species to apply this model on. This is described more detailed in the next subsections.

### 2.1.1 The Constant Rate Model

Assuming a **constant rate** (k = const. and a starting time $\mathrm{t_{start}}$), the equations 1 and 2 can be solved analytically:

$$m_s(t) = m_{s,final} + \left(m_s(t = t_{start,s}) - m_{s,final}\right)\, e^{-k_s(t-t_{start,s})} \tag{3}$$

$$m_{g,i}(t) = m_{g,i,final}\cdot\left(1 - e^{-k_i(t-t_{start,i})}\right) \tag{4}$$

$$if \quad t \le t_i: \quad m(t) = m(0) \tag{5}$$

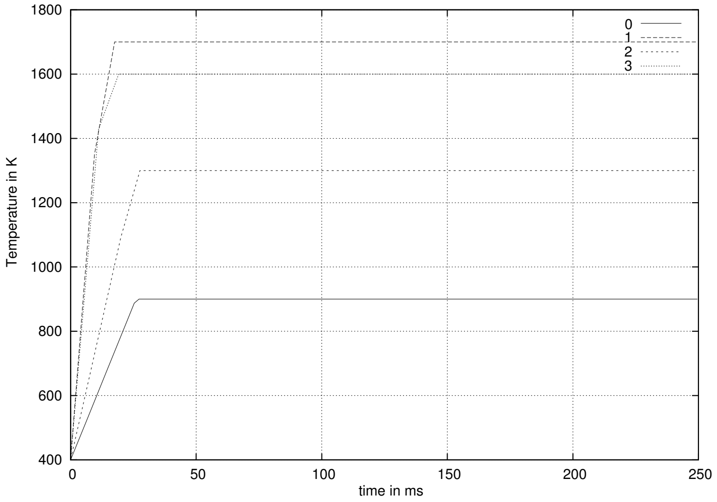


Figure 3: The temperature history.

This leads to advantage concerning the computational costs. On the other hand is this model completely independent from the temperature history. This is visualized in figure 4 where all fitted curves overlap each other.
The parameters to fit are for every species:

- the starting time ($t_{start,i}$), where the devolatilization begins
- the constant rate factor $k_i$
- the final yield (when more than one run)

This model is applied to all species.

### 2.1.2 The Arrhenius Model

The kinetic rate k can be also expressed with the **Arrhenius** equation:

$$\frac{dm_s}{dt} = A_s \cdot T(t)^{b_s} \cdot e^{-\frac{E_s}{T(t)}} \left(m_s - m_{s,final}\right) \tag{6}$$

$$\frac{dm_{g,i}}{dt} = A_i \cdot T(t)^{b_{g,i}} \cdot e^{-\frac{E_{g,i}}{T(t)}} \left(m_{g,i,final} - m_{g,i}\right) \tag{7}$$

This notation of the Arrhenius equation includes no gas constant R in the exponential term. So the activation energy (or here more precise activation temperature) has the unit Kelvin. This is so far an advantage as the fitted $E_i$ is independent of the used unit system (SI or cgs).
A second notation of the Arrhenius equation contains not the correction term $T(t)^{b_{g,i}}$:

$$\frac{dm_s}{dt} = A_s \cdot e^{-\frac{E_s}{T(t)}} \left(m_s - m_{s,final}\right) \tag{8}$$

$$\frac{dm_{g,i}}{dt} = A_i \cdot e^{-\frac{E_{g,i}}{T(t)}} \left(m_{g,i,final} - m_{g,i}\right) \tag{9}$$

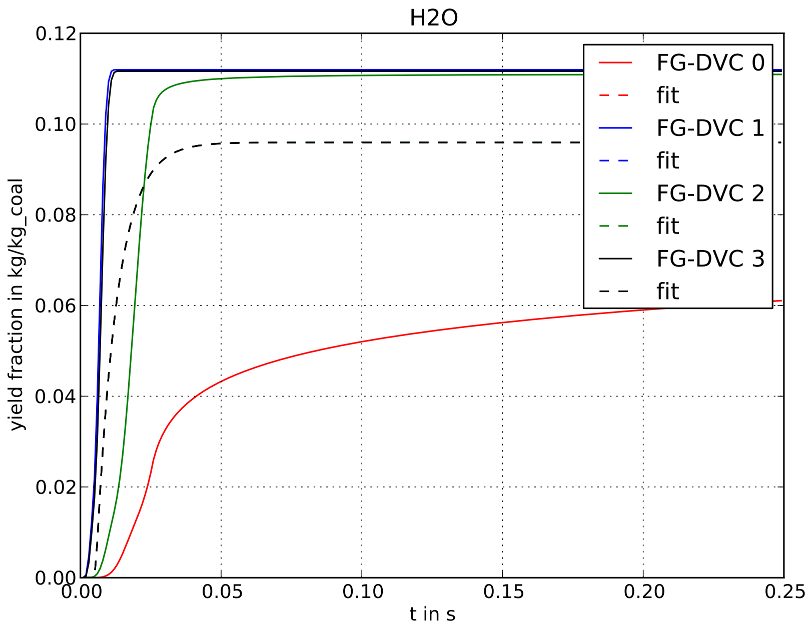


Figure 4: One fitting result (Yields) for the constant Rate Model. The observed species is water.

Unlike the constant rate model is the Arrhenius modeled rate influenced by the temperature. But the Arrhenius model can be used to express the evolve for all species and the final yields are also fixed. So the parameter to fit are here:

- the preexponentiation factor $A_i$
- the correction factor $b_i$
- the activation energy $E_i$
- the final yield (when more than one run)

This model is applied to all species.

The Arrhenius model leads to a good agreement in the yield and rate curves for a limited range of temperatures, figures 5, 6. The disadvantage is the temperature independent yield fraction, all integrals for the rate curves in figure 6 are the same. This leads to an imprecision as the yields show a dependency on the final temperature and heating rate.

### 2.1.3 The Kobayashi Model

Also the **Kobayashi** equation, also Two Competing Reaction Model, can be fitted, see equation 10. The optimization is carried out using the Arrhenius notation of equation 11 for $k_1$ and $k_2$.

$$\frac{m_v(t)}{m_{p,0} - m_a} = \int_0^t (\alpha_1 k_1 + \alpha_2 k_2) exp \left( - \int_0^t (k_1 + k_2) \, dt \right) \, dt \tag{10}$$

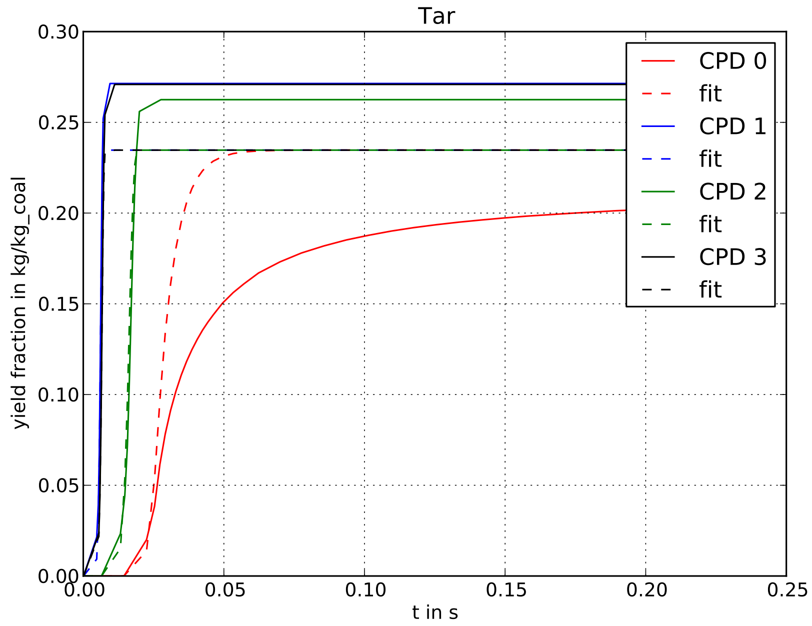


Figure 5: One fitting result (Yields) for the Arrhenius Model. The observed species is tar.

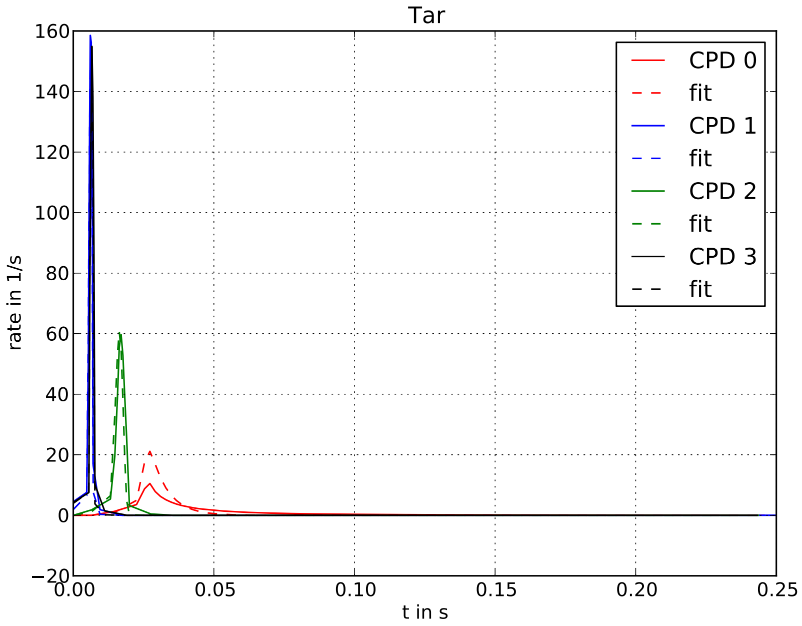


Figure 6: One fitting result (Rates) for the Arrhenius Model. The observed species is tar.

$$k_j = A_j e^{-\frac{E_j}{T(t)}} \qquad with\, j = 1{,}2 \tag{11}$$

The Kobayashi model can be applied only on the the overall, the total yields. The yield of individual species could be generated by multiplying the overall yield with the yield fraction $\frac{y_i}{y_{all}}$. But as the composition of the yields varies with the temperature history this factor also shows this dependency, which may leads to an imprecision when modeling the individual yields.
The final yields of this model are dependent on the temperature, see figures 7 and 8. The range of the yields are defined by the two weight factors $\alpha_1$ and $\alpha_2$. The $k_1$ models the reactions at lower temperatures (low $A_1$ and $E_1$), $k_2$ at higher temperatures (high $A_2$ and $E_2$). If the final temperature has very low values, the yields will converge to $\alpha_1$. If the temperatures will raise to infinity, the yields will be equal $\alpha_2$. So $\alpha_2$ is ever set equal one: $\alpha_2 = 1$. $\alpha_1$ is equal the amount of volatile matter in the daf coal. As the measurements, the proximate analysis of coal is based on, were carried out at very low heating rates compared with the ones occurring at gasification and combustion processes, the approximation $\alpha_1 = VM$ is an applicable and good assumption.

For the fitting procedure, the inner integral $\int_0^t (k_1 + k_2)\, dt$ is approximated by the Trapezoidal rule.

As it can be seen in figures 7 and 8, the higher temperatures (figure 3) lead to higher yields. But the influence of the temperature cannot be modeled that the dependency on temperature is completely the same as in the output of the more complex pyrolysis programs. So leads the higher temperature in case 1 compared to case 3 to a slightly higher yield in the output of `CPD` while the influence on the Kobayashi modeled result is greater.

### 2.1.4 The Distributed Activation Energy Model

The Distributed Activation Energy Model (DAEM) considers parallel first order kinetics over a specific range, described by a distribution function (F(E)). The equation used in PKP is:

$$m = m_{final} \left( 1 - \int_0^{\infty} exp \left[ -A_0 \cdot \int_{t_0}^{t_{final}} exp \left( -\frac{E}{T} \right) dt \right] F(E) \right) \tag{12}$$

As a distribution function, a Gaussian Distribution is used:

$$F(E) = \frac{1}{\sigma \cdot \sqrt{2\pi}} \cdot exp \left( -\frac{(E - E_0)^2}{2\sigma^2} \right) \tag{13}$$

So there are four parameters to optimize:

1. the preexponentiation factor $A_0$, which is here equal for all reactions.
2. $E_0$, defining the center of the Gaussian Distribution
3. and $\sigma$, which spcifies the flattening of the Distribution curve and its range
4. for multiple runs, the $m_{\text{final}}$ also has to be optimized

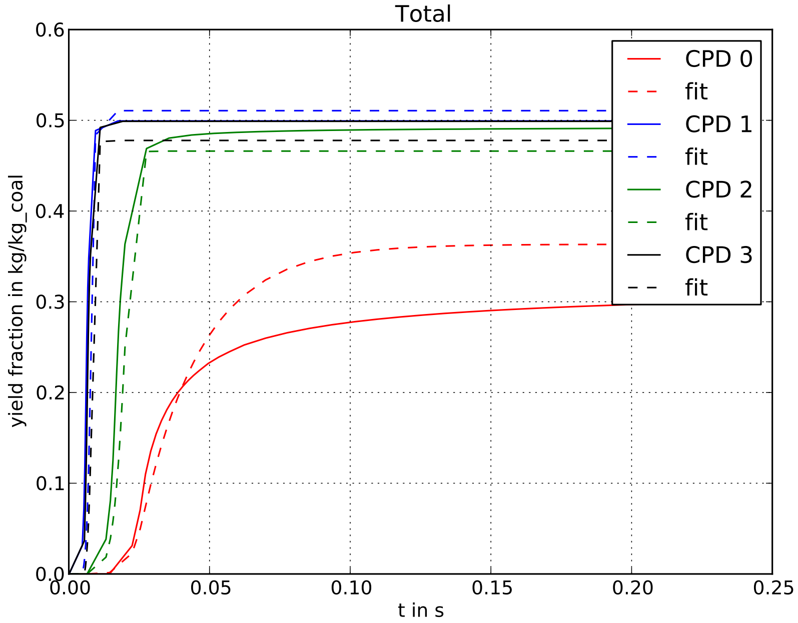


Figure 7: One fitting result (Yields) for the Kobayashi Model. The Kobayashi model just optimizes the overall yields.

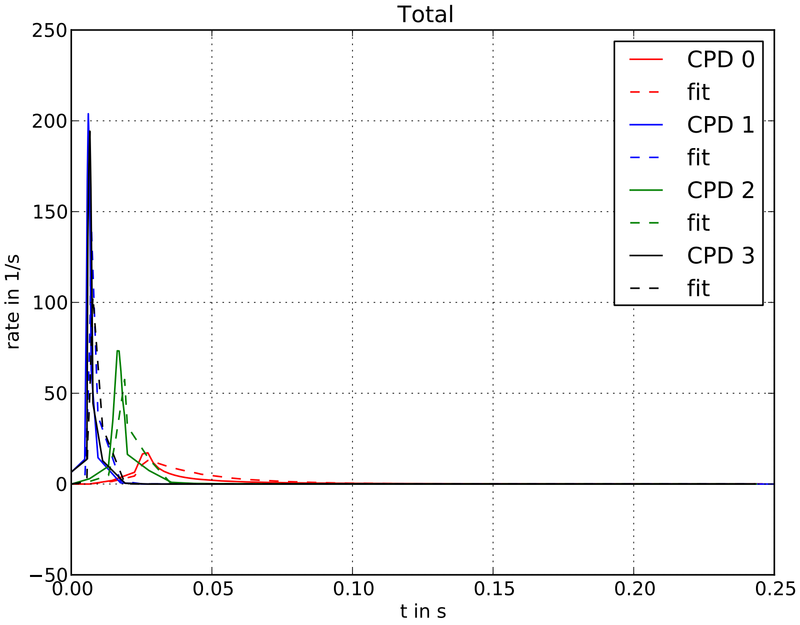


Figure 8: One fitting result (Rates) for the Kobayashi Model.

For solving the outer integral over dE, the Simpson tule is used. But for the nemuerical solution of the integral, the range has to be modified. As reported in the paper by Cai [3], the integration boundaries can be se to $\mathrm{E}_0 + 3 \cdot \sigma$ as the upper and $\mathrm{E}_0 - 3 \cdot \sigma$ as the lower limit. This range covers up to 99.73 % of the applied Gaussian Distribution. So for the further fitting of the devolatilization reaction, the prerequisite $\mathrm{E}_0 > 3 \cdot \sigma$ should be in force to achieve realistic results.
The inner integral $\int_{\mathrm{t}_0}^{\mathrm{t}_\mathrm{final}} \exp\left(-\frac{\mathrm{E}}{\mathrm{T}}\right) \mathrm{dt}$ was already simplyfied by setting $\mathrm{A}_0$ as a constant. In many papers [3, 2, 4, 5] a linear heating rate over the whole time range is assumed, so $\frac{\mathrm{dT}}{\mathrm{dt}} = \beta$. The transformation of the integral allows to integrate over the temperature. For such temperature integrals different analytical approaches exist [3, 2]. This method was also tested. But as this is only a very specific case of the operating conditions and even not faster than solving the integrals[9], this approach is furthernot considered. The double integral is solved numerically.[10]
The outer integral is solved over a specific number of activation energies. For each activation energy the inner integral is solved and the value for all time steps saved, using a compisite Trapezoidal rule[11]. All inner integrals are saved in a 2D-Array($\mathrm{t}_\mathrm{i}$,$\mathrm{E}_\mathrm{i}$). Each column contains all values of the inner integral for all time steps (the same activation energy), each line allinner integrals at the time $\mathrm{t}_\mathrm{i}$. After this array is calculated, the equation of the outer integral is used for all values of the current line of the matrix and afterwards this list is integrated over dE.

### 2.1.5 Fitting the Kinetic Equations

The fitting procedure is carried out with a `scipy-optimizer`[12] [13] and the `scipy.odeint`[14] package to minimize the residual $E(k,m_{fit})$ in the equation 14. For the structure of the whole fitting procedure see chapter 3. In equation 14 is $\mathrm{m}_\mathrm{out}$ the output of the devolatilization program `CPD` or `FG-DVC` . The optimization is carried out over all points reported in the output file of the pyrolysis program. The normalized weight factor parameters $\mathrm{a}_0$ and $\mathrm{a}_1$ in the equations 15 and 16 can both be defined by the user, the standard setting is for both one.

$$E(k,m_{fit}) = \omega_0 \int (m_{out}(t) - m_{fit}(k,t))^2 \, dt \; + \; \omega_1 \int (\dot{m}_{out}(t) - \dot{m}_{fit}(k,t))^2 \, dt \tag{14}$$

$$\omega_0 = \frac{a_0}{(max(m_{out}) - min(m_{out}))^2} \tag{15}$$

$$\omega_1 = \frac{a_1}{max(\dot{m}_{out}^2)} \tag{16}$$

[9] The reason might be that the very large analytical equations in Python (as an interpreting language) take more time than let the integral solve by a external library.

[10] There are also other approaches to avoid the double integration like in the paper by McGuiness et al. [6], but also these assumptions made here ($\sigma \rightarrow 0$ or $\frac{\mathrm{E}_0}{\mathrm{T}} \rightarrow \infty$) cannot be applied here.

[11] the `scipy.integrate.cumtrapz` module, http://docs.scipy.org/doc/scipy/reference/generated/scipy.integrate.cumtrapz.html

[12] *http://docs.scipy.org/doc/scipy/reference/optimize.html*

[13] The standard setting optimizer is `fmin`. The `leastsq` optimizer is the second choice.

[14] *http://docs.scipy.org/doc/scipy/reference/generated/scipy.integrate.odeint.html*

### 2.1.6 Pyrolysis Species- and Energy Conservation for CPD output

**Species Conservation** As the first step it has to be checked that the oxygen content in the generated yields (oxygen containing species $\mathrm{f_i}$ with $\mu_i^O$ oxygen) is less equal that the oxygen in the ultimate analysis:

$$UAO^{cpd,species\,output} = M_O \sum_i \frac{\mu_i^O f_i}{M_i} \leq UAO \tag{17}$$

The factor $\gamma$ (18) tells if the outputted yields contain less oxygen than reported in the Ultimate Analysis (UA) ($\gamma > 1$) or if they are equal ($\gamma = 1$). For the case that $\gamma < 1$, the oxygen containing yields have to be decreased by using equation 19, while the amount of the other species have to be increased to conserve the conserve the amount of volatile matter (equation 20). In this case, the tar will contain no oxygen. The yield of $N_2$ is equal to the UA of Nitrogen.

$$\gamma = \frac{UAO}{UAO^{cpd,species\,output}} \tag{18}$$

$$f_i^{new} = \gamma \cdot f_i \tag{19}$$

$$f_{oth}^{new} = f_{oth} + (1-\gamma)\sum_i f_i \tag{20}$$

For the case $f_{N_2} < f_{other}$, the remaining part is assigned to $CH_4$:

$$f_{CH_4}^{new} = f_{CH_4} + (f_{oth}^{new} - f_{N_2}) \tag{21}$$

Now the composition of tar can be calculated. For each element C,H,O, the following equation 22 can be used, assuming a tar composition of $C_nH_mO_p$. $M_j$ is the atom weight of the element j, $\mu_i^j$ the number of atoms of $j$ in the species $i$.

$$\frac{UA_j}{M_j} = \mu_{tar}^j \frac{f_{tar}}{M_{tar}} + \sum_i \mu_i^j \frac{f_i}{M_i} \tag{22}$$

**Energy Conservation** The Dulong formula is used, if the higher heating value (HHV) of the coal is not known:

$$HHV = 32.79 \cdot UAC + 150.4 \cdot (UAH - UAO/8) + 9.26 \cdot UAS + 4.97 \cdot UAO + 2.42 \cdot UAN \tag{23}$$

where UAC, UAH, UAO, UAS and UAN are the value of the ultimate analysis for carbon, hydrogen, oxygen, sulfur and nitrogen. The result has the unit of $\frac{MJ}{kg_{coal,asrecieved}}$.

Afterwards, the HHV (entered by the user or calculated with the Dulong formula) for the coal as received is related to the dry ash-free (daf) state (equation 24). This new HHV is used to get the lower heating value for a daf state, equation 25. In this equation, $r_{H_2O}$ is the latent heat of water.

$$HHV_{daf} = \frac{HHV_{ar}}{PAVM + PAFC} \tag{24}$$

$$LHV_{daf} = HHV_{daf} - \frac{M_{H_2O}}{2 \cdot M_H} \cdot UAH \cdot r_{H_2O} \tag{25}$$

Regarding the combustion of the raw coal (equation 26), the energy balance can be written as in equation 27.

$$C_xH_yO_zN_w + (x+y/4-z/2)O2 \rightarrow xCO2 + y/2H2O + w/2N2 \tag{26}$$

$$Q_{react} = LHV_{raw} \cdot M_{daf} = h_{f,raw} + (x+y/4-z/2)h_{f,O_2} - xh_{f,CO_2} - y/2h_{f,H_2O} - w/2h_{f,N_2} \tag{27}$$

Using equation 27, the heat of formation of the raw molecule ($h_{f,raw}$) can be calculated.

The heat of formation for tar is based on the equation 28, implying, that no heat is produced or absorbed during the devolatilization process.

$$C_xH_yO_zN_w \rightarrow \nu_{char}C_{(s)} + \nu_{tar}C_nH_mO_p + \sum_i \nu_iM_i \tag{28}$$

The stoichiometric coefficient of each species can be calculated from the volatile yield expressed as mass fraction:

$$\nu_i = \frac{f_iM_{raw}}{M_i} \tag{29}$$

Making the energy balance for equation 28 with $Q_{react} = 0$, the heat of formation for tar is:

$$\nu_{tar}h_{f,tar} = h_{f,raw} - \nu_{char}h_{f,char} - \sum_i \nu_ih_{f,i} \tag{30}$$

Another method is to assume a heat of formation for tar equal zero (e.g. if there is only a very low yield of tar), and calculate the heat of pyrolysis:

$$-Q_{pyro} \cdot M_{raw} = h_{f,raw} - \nu_{char}h_{f,char} - \sum_i \nu_ih_{f,i} \tag{31}$$

Where $Q_{pyro}$ is the heat of pyrolysis per unit of mass of daf. It is positive if heat is required for breaking coal structure bounds. Generally, it is expressed in terms of volatile matter:

$$Q_{pyro}^{vm} = \frac{Q_{pyro}}{1-f_{char}} \tag{32}$$

### 2.1.7 Pyrolysis Species and Energy Conservation for FG-DVC output

**Species Conservation** As in most of the CFD applications the combustion of some FG-DVC output species like HCN, COS or Olefins are not implemented. Only the species Char, Tar, CO, $CO_2$, $H_2O$, $CH_4$ and $H_2$ are further considered. The nitrogen is merged into the tar. So the amount of tar is calculated by the using equation 33, where the sum contains the species Char, CO, $CO_2$, $H_2O$, $CH_4$ and $H_2$.

$$f_{Tar} = 1 - \sum_i f_i \tag{33}$$

Applying the equation 22 for all the elements j (Carbon, Hydrogen, Nitrogen, Oxygen), the composition of tar (its stoichiometric coefficients) is calculated.

**Energy Conservation** Applying the energy balance on the combustion reaction of the devolatilization yields (for the case of a non-heat producing/consuming pyrolysis process), the following reaction equation is satisfied:

$$LHV_{daf} = H_{f,Tar} \cdot f_{Tar} + \sum_i H_{f,i} \cdot f_i \tag{34}$$

The LHV is calculated based on the HHV using the same equations as in chapter 2.1.6. The $H_{f,i}$ are calculated with the following equations 35 to 38, making an energy balance for every of the pyrolysis yields. So the heat of formation for tar can be calculated from equation 34, as all other parameters are known.

$$H_{f,Char} = \left((h_{f,Char} + h_{f,O_2} - h_{f,CO_2}) \cdot f_{Char}\right) \cdot M_C^{-1} \tag{35}$$

$$H_{f,H_2} = \left((h_{f,H_2} + \frac{1}{2} \cdot h_{f,O_2} - h_{f,H_2O}) \cdot f_{H_2}\right) \cdot M_{H_2O}^{-1} \tag{36}$$

$$H_{f,CH_4} = ((h_{f,CH_4} + 2 \cdot h_{f,O_2} - h_{f,CO_2} - 2 \cdot h_{f,H_2O}) \cdot f_{CH_4}) \cdot M_{CH_4}^{-1} \tag{37}$$

$$H_{f,CO} = \left((h_{f,CO} + \frac{1}{2} \cdot h_{f,O_2} - h_{f,CO_2}) \cdot f_{CO}\right) \cdot M_{CO}^{-1} \tag{38}$$

The $H_{f,Tar}$ with the unit $\frac{J}{kg}$ is transformed back into $\frac{J}{kmol}$ by multiplying with the molecular mass of tar.

To calculate the heat of formation for tar, the tar combustion can be regarded, as the tar composition is known:

$$C_nH_mO_pN_k + \nu_{O_2}O_2 \rightarrow \nu_{CO_2}CO_2 + \nu_{H_2O}H_2O + \nu_{N_2}N_2 \tag{39}$$

This leads to the balance, where the $h_{f,Tar}$ can be calculated:

$$H_{f,Tar} = h_{f,Tar} + \nu_{O_2}h_{f,O_2} - \nu_{CO_2}h_{f,CO_2} - \nu_{H_2O}h_{f,H_2O} - \nu_{N_2}h_{f,N_2} \tag{40}$$

# 3 Programs Structure

The files, the directory must contain, are listed in chapter 1.1. The Python script *Pyrolysis.py* accesses to all the defined classes and controls the whole PKP program. So PKP can be launched the simplest way by starting *Pyrolysis.py*.
The figure 9 gives a overview of the program structure.

The whole program structure is visualized in figure 9, the fitting structure in figure 10. For a detailed documentation for each class and its method, see the document *PKP Code Documentation*.
All required Python packages are also listened in this document.[15]

In figure 9 the external programs are colored in red, the PKP classes are in green, marked below with the class name or the relevant file name. Inputted, transfered or outputted data is in black.
So as first the specific user information about the operating conditions and the coal properties are imported using the reading classes contained in *Information Files.py*.
Parts of this information are further used to write the input file for the `FG-DVC` coal generator[16], which is launched afterwards to generate the `FG-DVC` coal input files.

Afterwards the pyrolysis programs are launched automatically by PKP. The file *IN.dat* is an input file for `CPD` telling how the output files should be named. So the content in *IN.dat* shouldn't be changed. The central input file for `FG-DVC` is the *instruct.ini* which contains all information about the location of the coal file, the operating conditions and the numerical parameters. This file was generated by PKP before launching. The second input file for `FG-DVC` contains the temperature history. This file (named *tTHistory.txt*, located in the `FG-DVC` main directory) was also written by PKP before each run.
The output files of the pyrolysis programs are read by their specific **_Result* classes (e.g. *CPD_Result*). These classes save the information from the output files, contain a dictionary with the information which column contains which species and physical variable. The **_Result* classes also process the specific information (e.g. the modifications described in chapter 1.2, or the transformation from ms into s in the `CPD` output). Converting this pyrolysis program specific output shape into a general one has two main advantages. As this 'standard shaped' information are passed to the following fitting data class *Fit_one_run*, is the fitting procedure working for the output of all pyrolysis programs the same way. The second advantage is, that for a modified output of a new version of a pyrolysis program just a new **_Result* class needs to be added, modified for the new output shape. Using the dictionaries has the advantage that the code is better readable, more robust against modifications and more general purpose. So in the program the yield of tar can be requested with *self.Yield('Tar')* instead of using any number. A species must not have the same index in the output of the **_Result* classes, which might also not work as the pyrolysis programs have different number of species in the yields (`FG-DVC` has 15 species in the yields, `CPD` eight).

The fitting procedure is carried out using three classes: a Least Squares Estimator, a

[15] For Windows it is easier to install Python(x,y) (https://code.google.com/p/pythonxy/) which includes all these packages than install each of them manually.

[16] If chosen this way by the user. Otherwise also a selected `FG-DVC` library coal can directly be used to run `FG-DVC`. Then this step is not done.

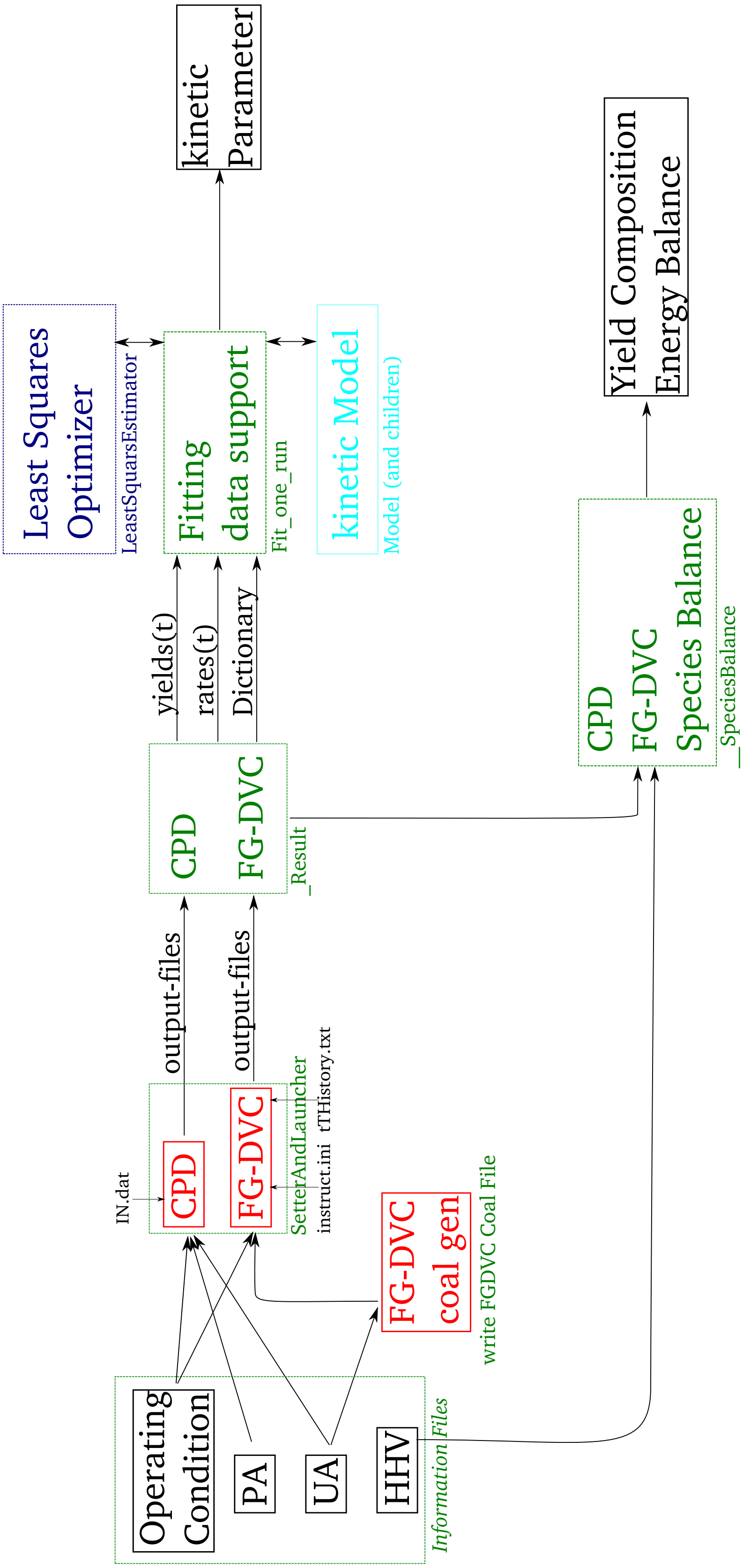


Figure 9: Overview of the program. The Optimization procedure is shown detailed in figure 10.

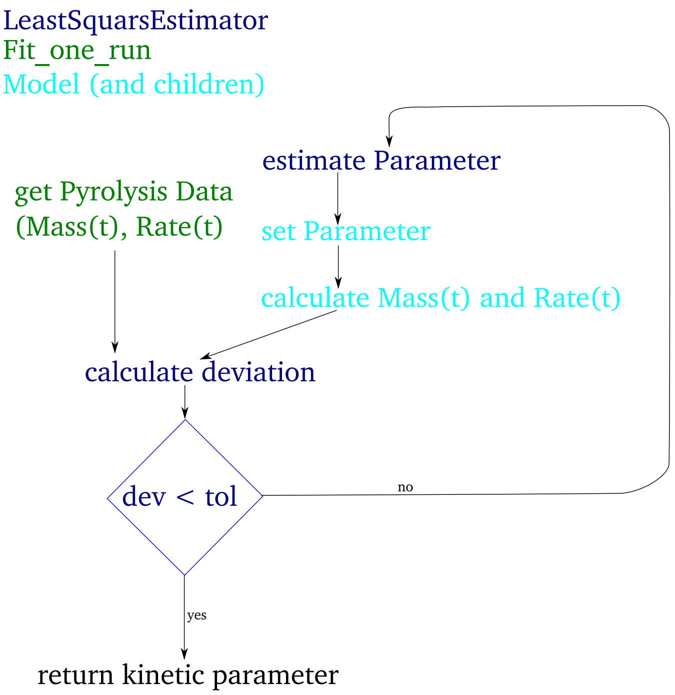


Figure 10: The optimization structure. The three classes *LeastSquarsEstimator*, *Fit_one_run* and *Model* are marked with different colors like in Figure 9. The Deviation is calculated with equation 14.

kinetic model and a class for the pyrolysis program results, see figure 9 and the more detailed figure 10. Depending on the selected kinetic model and its parameter to fit one of the children classes (e.g. constant Rate, of the parent class model) is selected. As the initial guess of the parameter[17] is in the physical realistic range, this problem can be fitted with a gradient based optimizer. The optimizer class calculates the deviation, equation 14 using the data from the *Fit_one_run* class[18] and the calculated yields of the model applying the current kinetic parameter. The deviation allows the optimizer a new estimation of the parameter. This estimated parameter are passed to the kinetic model class, which sets these parameter for a further use of calculating the mass over time. This loop is repeated until the deviation is lower than the setted tolerance.

The species and energy balance is carried out with specific classes for `FG-DVC` and `CPD`. As these are straight forward analytical calculations, chapters 2.1.6 and 2.1.7, these classes just needs to be initialized with the right input parameter to output afterwards the results applying their private methods, see the *PKP Code Documentation*.

[17] defined in the code
[18] which provides the pyrolysis program results

# References

[1] Advanced Fuel Research in 87 Church Street, East Hartford, CT 06108 USA. *THE FG-DVC COAL PYROLYSIS MODEL, USER'S GUIDE, Version 8.2.2*, 2010.

[2] J. Cai, C. Jin, S. Yang, and Y. Chen. Logistic distributed activation energy model – Part 1: Derivation and numerical parametric study. *Bioresource Technology*, (102):1556–1561, 2011.

[3] J. Cai and R. Liu. New distributed activation energy model: Numerical solution and application to pyrolysis kinetics of some types of biomass. *Bioresource Technology*, (99):2795–2799, 2008.

[4] J. Cai, S. Yang, and L. Tao. Logistic distributed activation energy model – Part 2: Application to cellulose pyrolysis. *Bioresource Technology*, (102):3642–3644, 2011.

[5] L. Gašparovič, J. Labovskỳ, J. Markoš, and L. Jelemenskỳ. Calculation of Kinetic Parameters of the Thermal Decomposition of Wood by Distributed Activation Energy Model (DAEM). *Chem. Biochem. Eng*, (26):45–53, 2012.

[6] M. McGuinness, E. Donskoi, and D. McElwain. Asymptotic Approximations to the Distributed Activation Energy Model. *Applied Mathematics Letters*, (12):27–34, 1999.