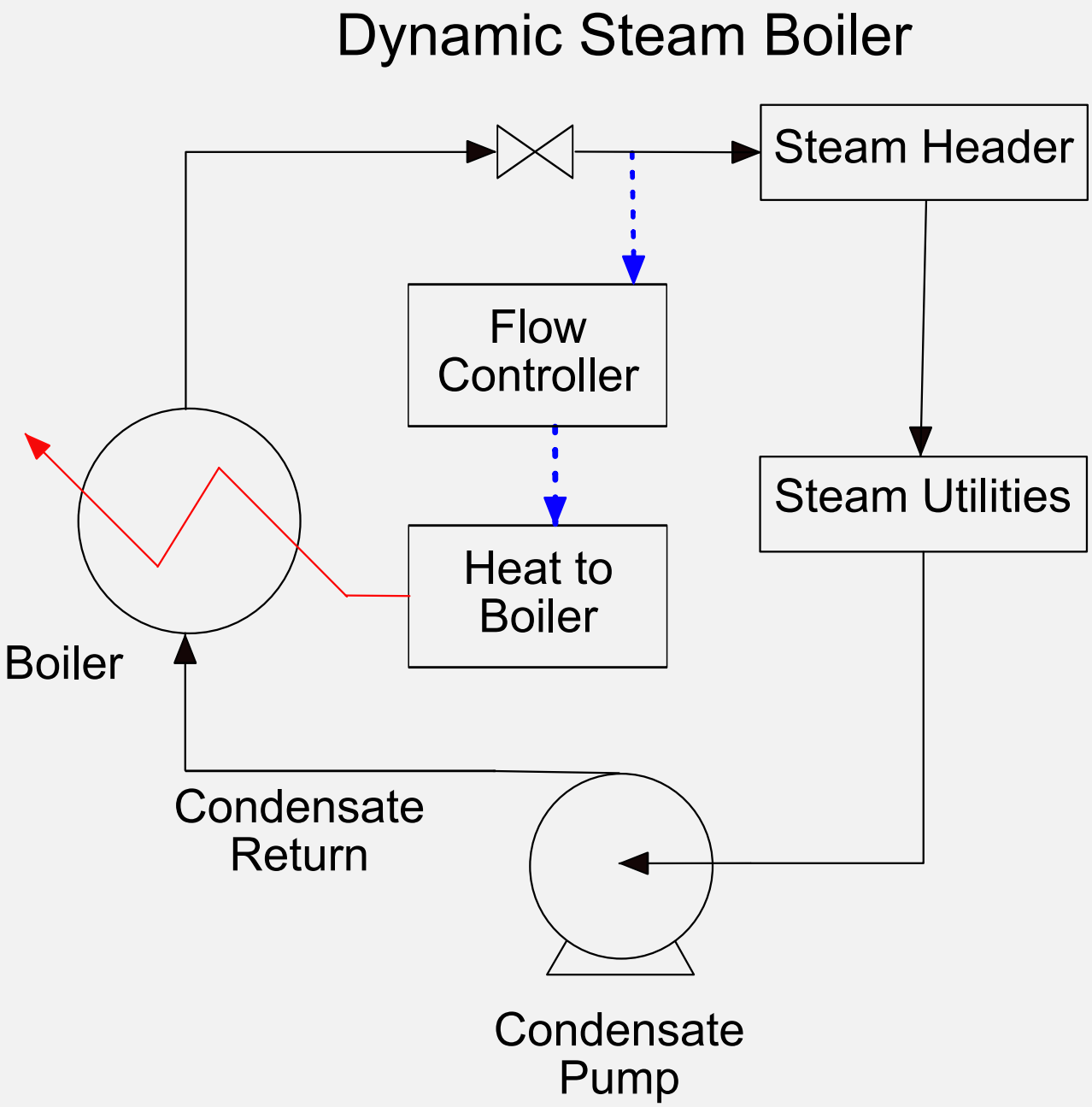
Idealized Steam Boiler Dynamics

CoolProp has the ability to do flashes where one of the intensive properties is a density, but only for systems that contain a single compound. In addition isenthalpic and isentropic flashes are much faster for single component systems since CoolProp handles these natively, while Math Minion adds a search mechanism to find a result for multicomponent systems.

This model demonstrates the use of a denisty-enthalpy flash for steam in a highly idealized dynamic simulation of a steam boiler.



Steam is raised in a boiler and accumulates in its steam chest, before flowing through a valve into a steam utilities header. From there it is used in some sort of utilities, resulting in condensate, that is pumped back up to the boiler pressure and arbitrarily assumed to be 80 C. The condensate flow is assumed to be proportional to the holdup in the condensate tank, while a proportional integral controller regulates the heat

flow to the boiler to meet a steam flow set point.

The model integrates four values, with the initial values set in expression **initialY**:

boiler holdup (95 kg)
condensate holdup (50 kg)
boiler energy (120,000 kJ)
steam flow controller accumulated error (0 unitless)

The **boilerFlash** uses the formula

boilerHoldup / water.mwt / SteamChest

To determine the density of the steam in the steam chest. A dummy flash **water** is used merely to obtain the molecular weight of water, while the **SteamChest** expression specifies the volume of the steam chest as 3 m^3. This formula is entered, rather counterintuitively, in the **T or P** field, while the boiler energy holdup integrated by the **ode** tool and represented by the **boilerH** expression goes in the **2nd Prop** field. The result is a density/enthalpy flash that determines the steam properties, including pressure.

The flow of steam through the valve to the steam header is calculated from a form of the flow coefficient equation, using the pressure difference between the calculated boiler pressure and the specified steam header pressure of 700 kPa.

The valve is assumed to be isenthalpic and so the boiler vapour enthalpy can be used with the specified **utilityP** pressure to determine the steam header properties in flash **steamHeader**.

The properties of the condensate return to the boiler, that is after the pump, are calculated in flash **utilityCond** using the boiler pressure and the assumed 80 C temperature. The flow rate is a simple function of the condensate hold up:

ode.y.condHoldup * 1 "1/min"

With these properties, the derivatives for **ode** can be calculated.

The derivitive for boiler mass holdup is simply the difference between the steam produced and the return condensate, which is calculated in expression **dBoilerMass** as:

utilityCond.massf - steamHeader.massf

Similarly the condensate holdup derivative is just the negative of the above difference:

steamHeader.massf - utilityCond.massf

which is calculated in **dCondMass** for clarity, but just using **-dBoilerMass** would have been sufficient.

A heat balance around the boiler provides the boiler energy derivative, by subtracting the steam header energy flow, again assuming the valve to be isentropic, from the supplied boiler heat plus the energy of the condensate return. This is done in expression **dBoilerEnergy**:

boilerHeat + utilityCond.hFlow - steamHeader.hflow

Finally the error signal for the flow controller calculation is taken as the difference between the desired flow, 3000 kg/h, and the actual flow divided by a scaling factor. This is done in **FlowSigError** as:

(3000 "kg/h" - steamFlow) / 6000 "kg/h"

The error derivative for the integral portion of the controller is entered directly in the table of derivatives in **derivatives** as:

flowErrorSig / 1 s

The resulting heat flow to the boiler is calculated from the controller formula:

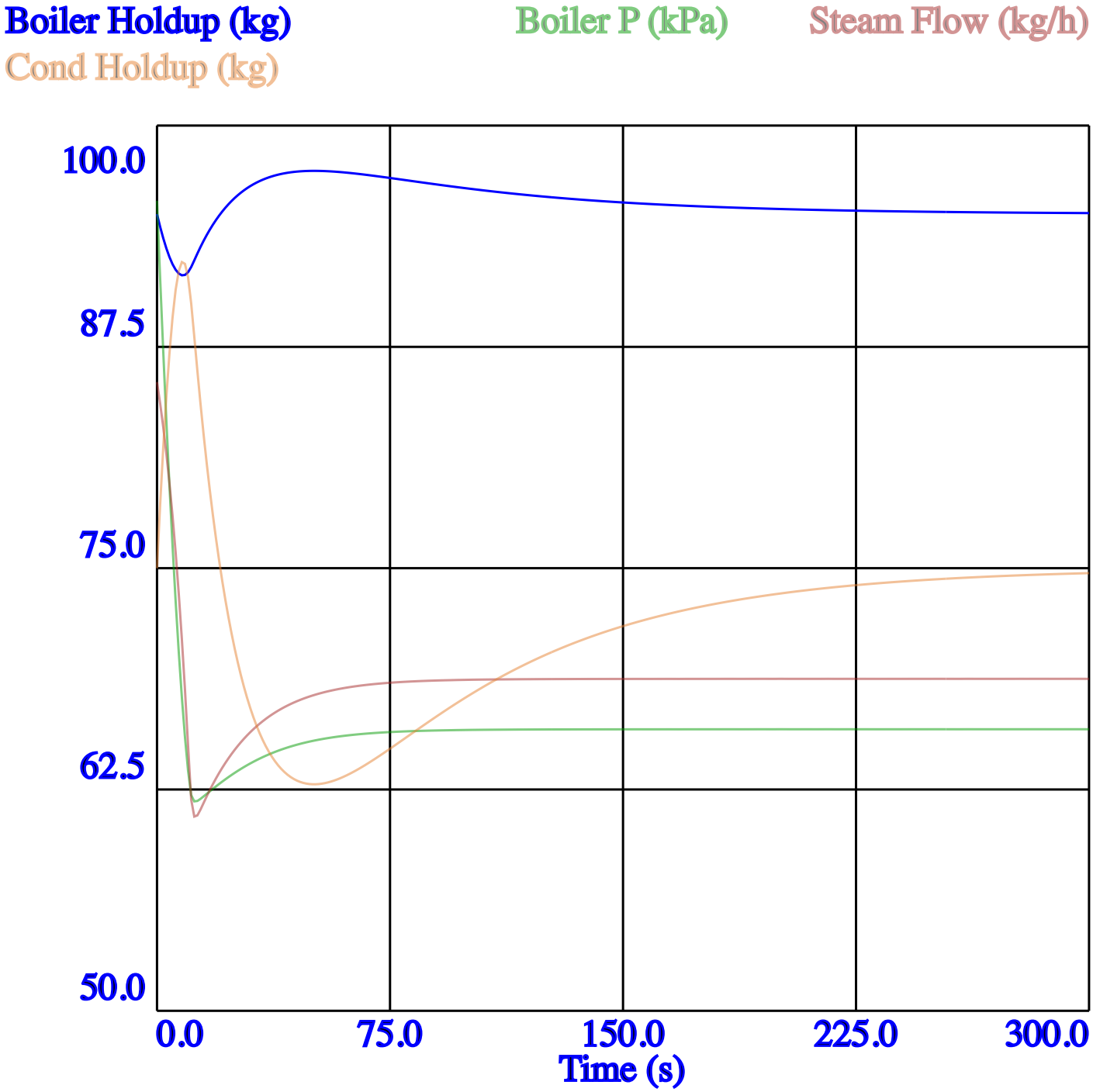
```
5000 kw *
        {max 0, {min 1, FlowErrorSig * 3 +
        ode.y.errorAccum * 0.2}}
```

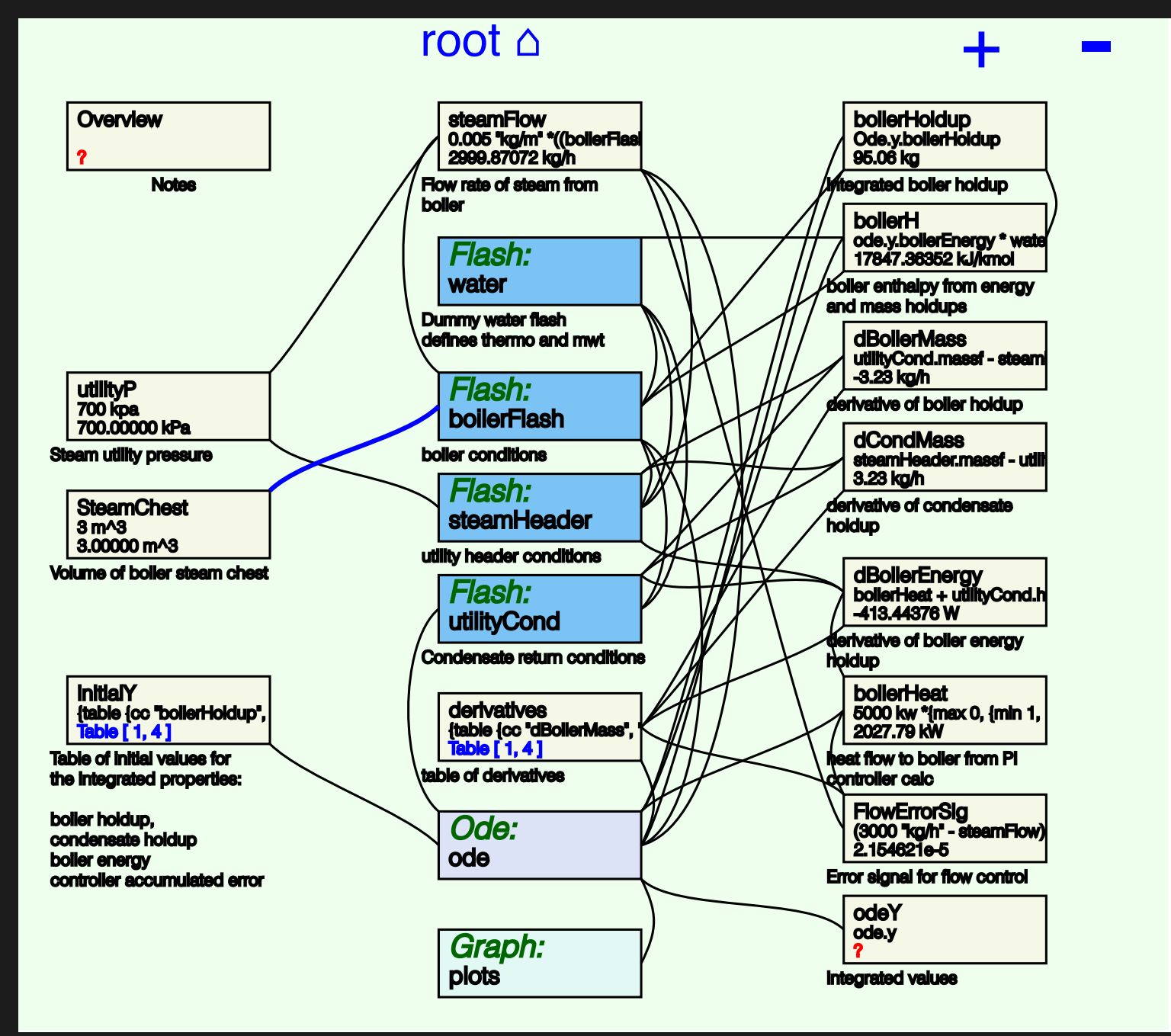
with the output signal using a proportionality constant of 3 and an integral constant of 0.2. The signal is constrained to the range 0 to 1 and multiplied by a heat flow of 5000 kw.

The **ode** tool records time, boiler holdup, boiler pressure, steam flow and condensate hold up and these are plotted in the **plots** with time on the x axis. Tap on the y axis in **plots** to cycle through the various lines. A static image is presented below:

Boiler Holdup 50 - 100 kg
Condenser Holdup 45-55 kg
Boiler Pressure 500 - 1500 kpa
Steam Flow 0 - 8000 kg/h

utilityP	700 kpa
SteamChest	3 m^3
initialY	{table {cc "boilerHoldup", "condHoldup", "boilerEnergy", "errorAccum" }, 95 kg, 50 kg, 120000 kj, 0 }
plots	


Boiler Holdup (kg)
Boiler P (kPa)
Steam Flow (kg/h)
Cond Holdup (kg)
100.0
87.5
75.0
62.5
50.0
0.0
75.0
150.0
225.0
300.0
Time (s)


root ⌂
Overview
?
Notes
steamFlow
0.005 "kg/m" *((boilerFlas
2999.87072 kg/h
Flow rate of steam from boiler
bollerHoldup
Ode.y.bollerHoldup
95.06 kg
Integrated boiler holdup
bollerH
ode.y.bollerEnergy * wate
17847.38352 kJ/kmol
boiler enthalpy from energy and mass holdups
Flash:
water
Dummy water flash
defines thermo and mwt
dBollerMass
utilityCond.massf - steam
-3.23 kg/h
derivative of boiler holdup
utilityP
700 kpa
700.00000 kPa
Steam utility pressure
Flash:
boilerFlash
boiler conditions
dCondMass
steamHeader.massf - utili
3.23 kg/h
derivative of condensate holdup
SteamChest
3 m^3
3.00000 m^3
Volume of boiler steam chest
Flash:
steamHeader
utility header conditions
dBollerEnergy
bollerHeat + utilityCond.h
-413.44376 W
derivative of boiler energy holdup
Flash:
utilityCond
Condensate return conditions
initialY
{table {cc "bollerHoldup",
Table [1, 4]
Table of initial values for the integrated properties:
boiler holdup,
condensate holdup
boiler energy
controller accumulated error
derivatives
{table {cc "dBollerMass", '
Table [1, 4]
table of derivatives
bollerHeat
5000 kw *{max 0, {min 1,
2027.79 kW
heat flow to boiler from PI controller calc
Ode:
ode
FlowErrorSig
(3000 "kg/h" - steamFlow)
2.154621e-5
Error signal for flow control
odeY
ode.y
?
Integrated values
Graph:
plots
Info
Sessions
Console
Undo
Redo
Units