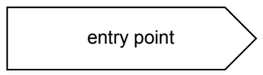
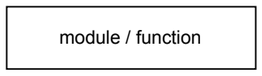
Functional Structure of Plant Model

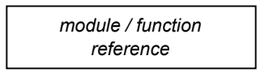
KEY



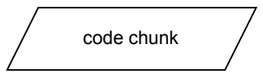
Entry point
— module / function that calls rest of functions



module / function
— functional component of model, may call other functions

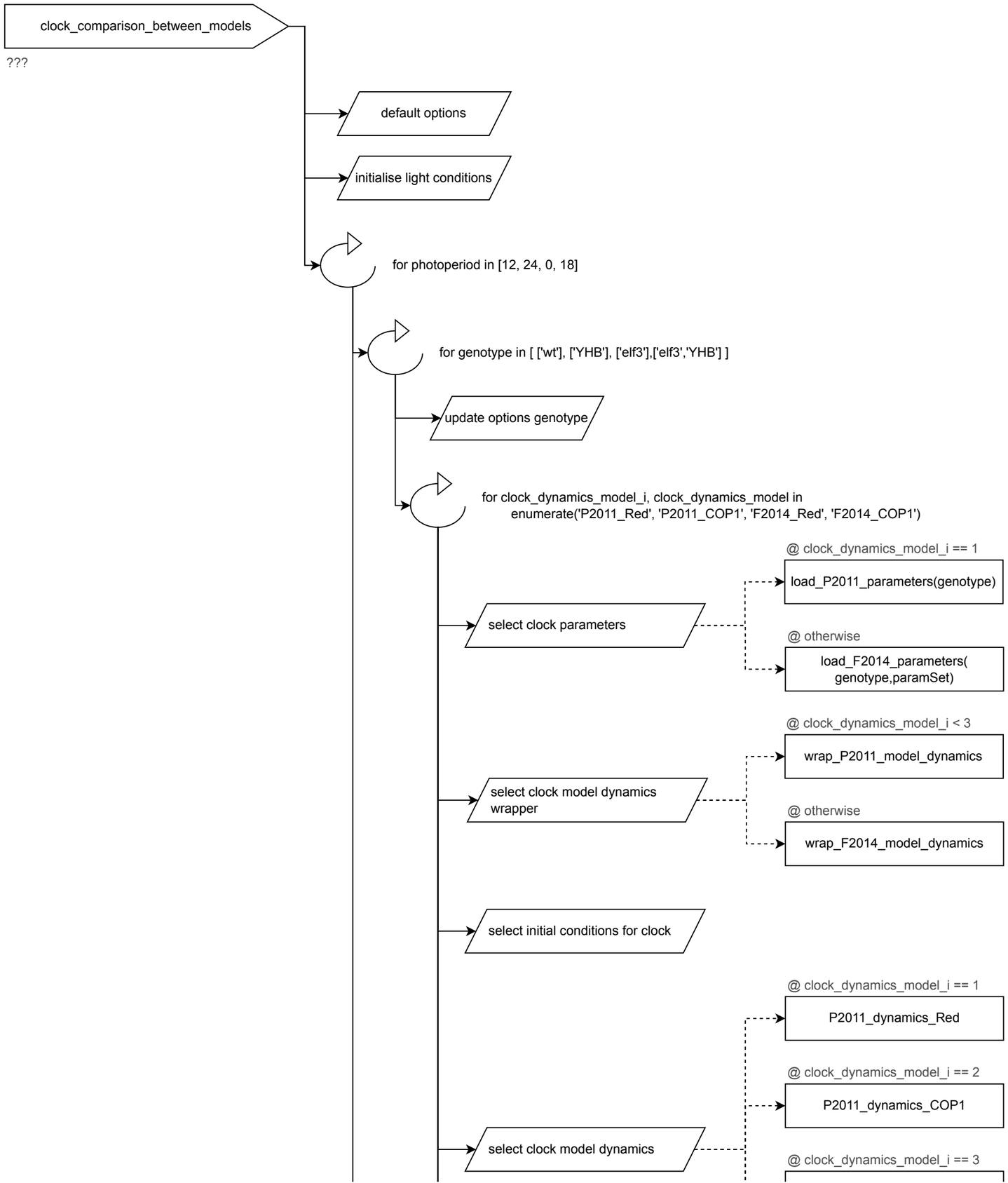


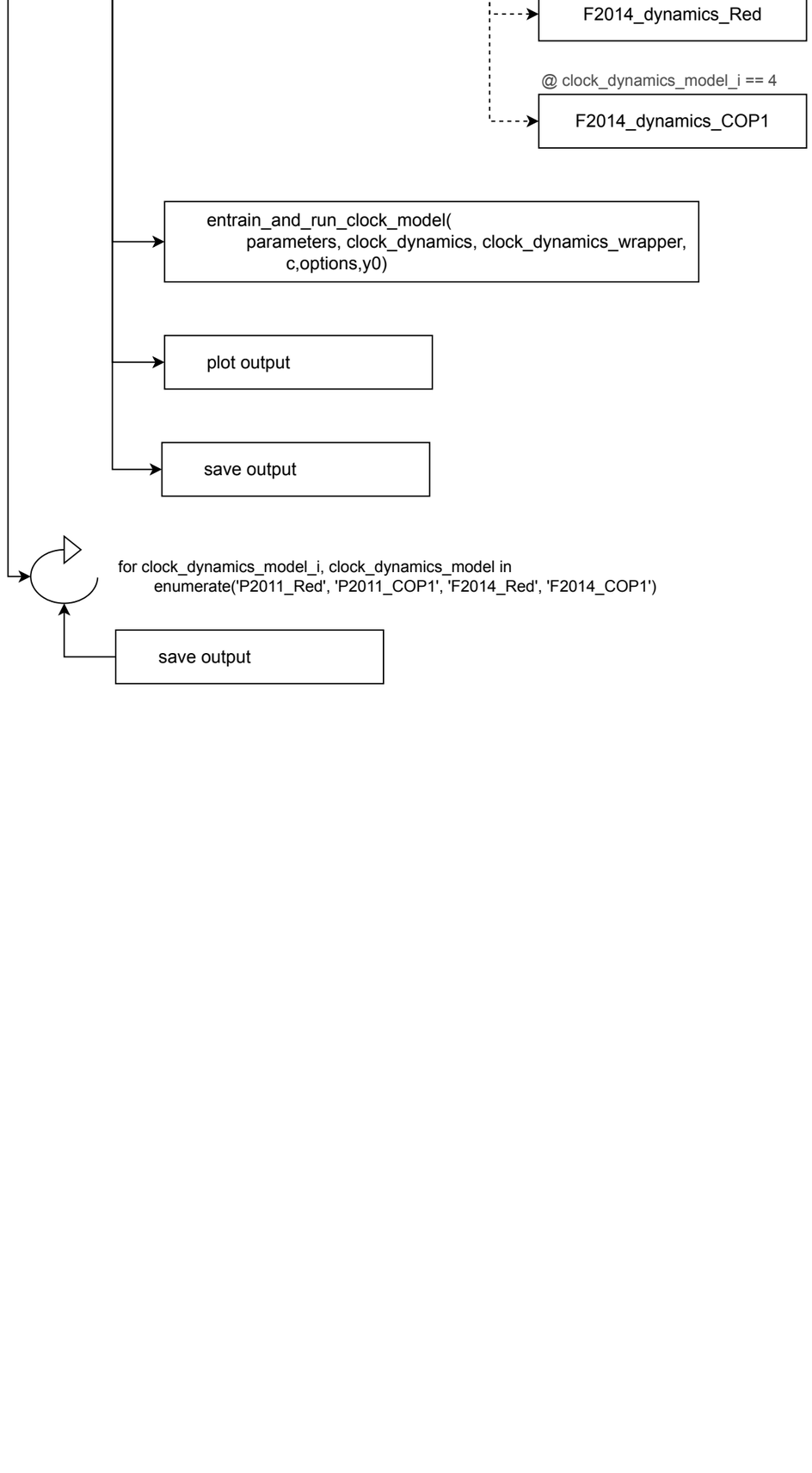
reference
— placeholder / reference for module / function unpacked elsewhere in document

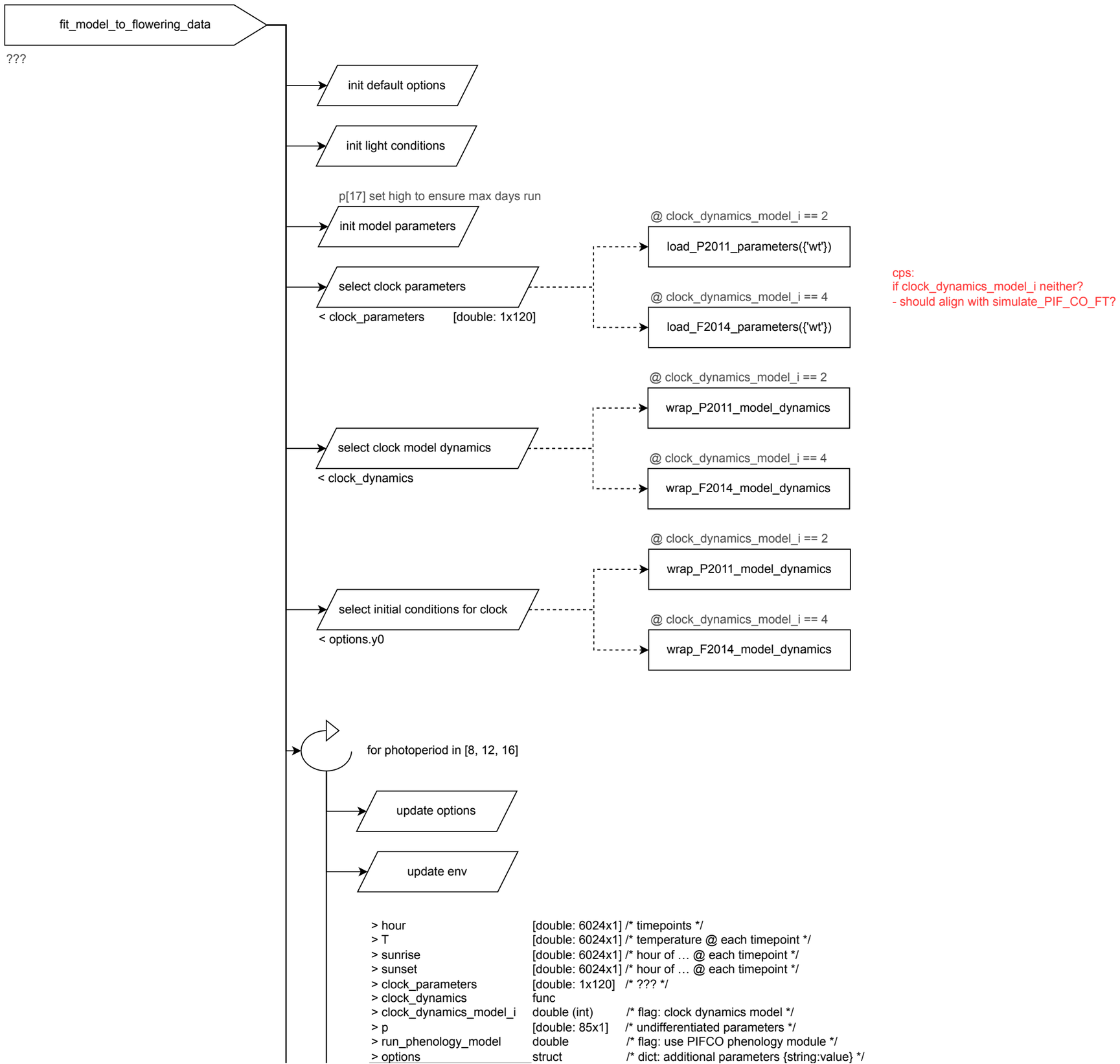


Chunk of code / logic
— logical chunk of code

Entry Points (1 / 4)

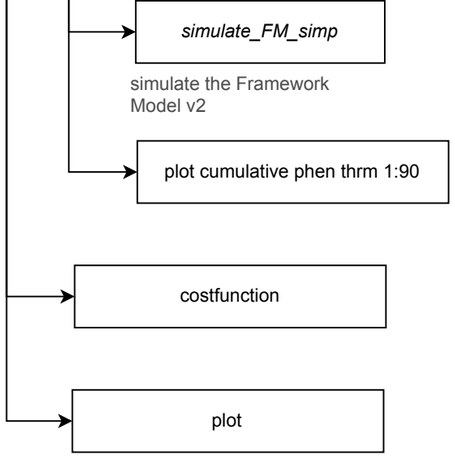


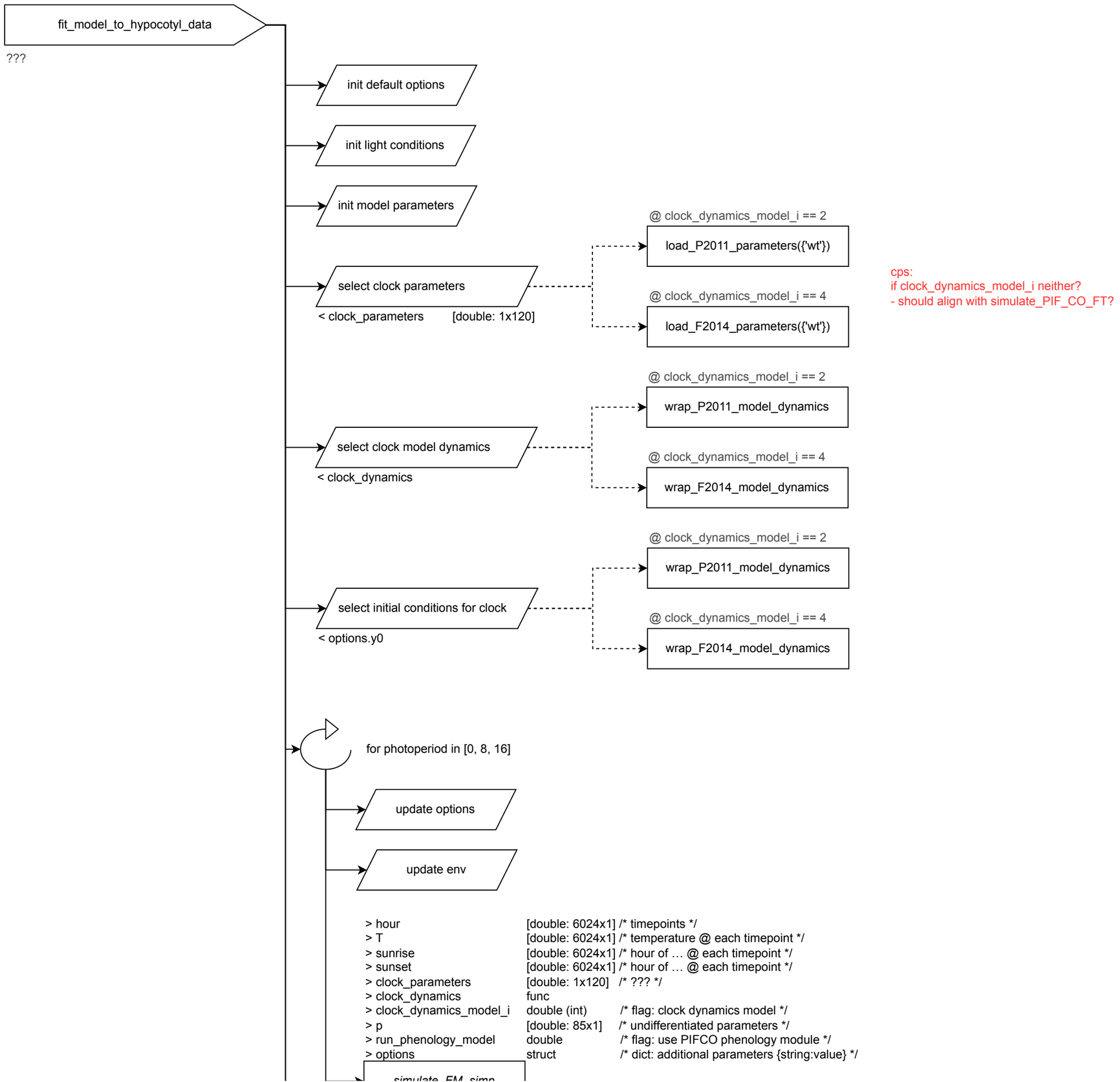

F2014_dynamics_Red
@ clock_dynamics_model_i == 4
F2014_dynamics_COP1
entrain_and_run_clock_model(
parameters, clock_dynamics, clock_dynamics_wrapper,
c,options,y0)
plot output
save output
for clock_dynamics_model_i, clock_dynamics_model in
enumerate('P2011_Red', 'P2011_COP1', 'F2014_Red', 'F2014_COP1')
save output

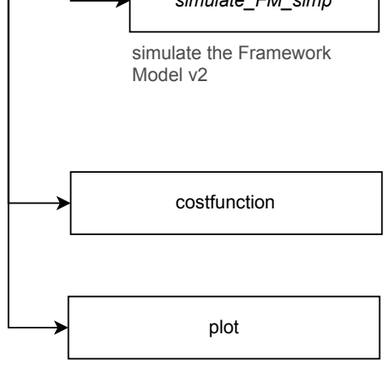


cps:
if clock_dynamics_model_i neither?
- should align with simulate_PIF_CO_FT?

```
> hour                  [double: 6024x1] /* timepoints */
> T                     [double: 6024x1] /* temperature @ each timepoint */
> sunrise               [double: 6024x1] /* hour of … @ each timepoint */
> sunset                [double: 6024x1] /* hour of … @ each timepoint */
> clock_parameters      [double: 1x120]  /* ??? */
> clock_dynamics        func
> clock_dynamics_model_i double (int)    /* flag: clock dynamics model */
> p                     [double: 85x1]   /* undifferentiated parameters */
> run_phenology_model   double           /* flag: use PIFCO phenology module */
> options               struct           /* dict: additional parameters {string:value} */
```

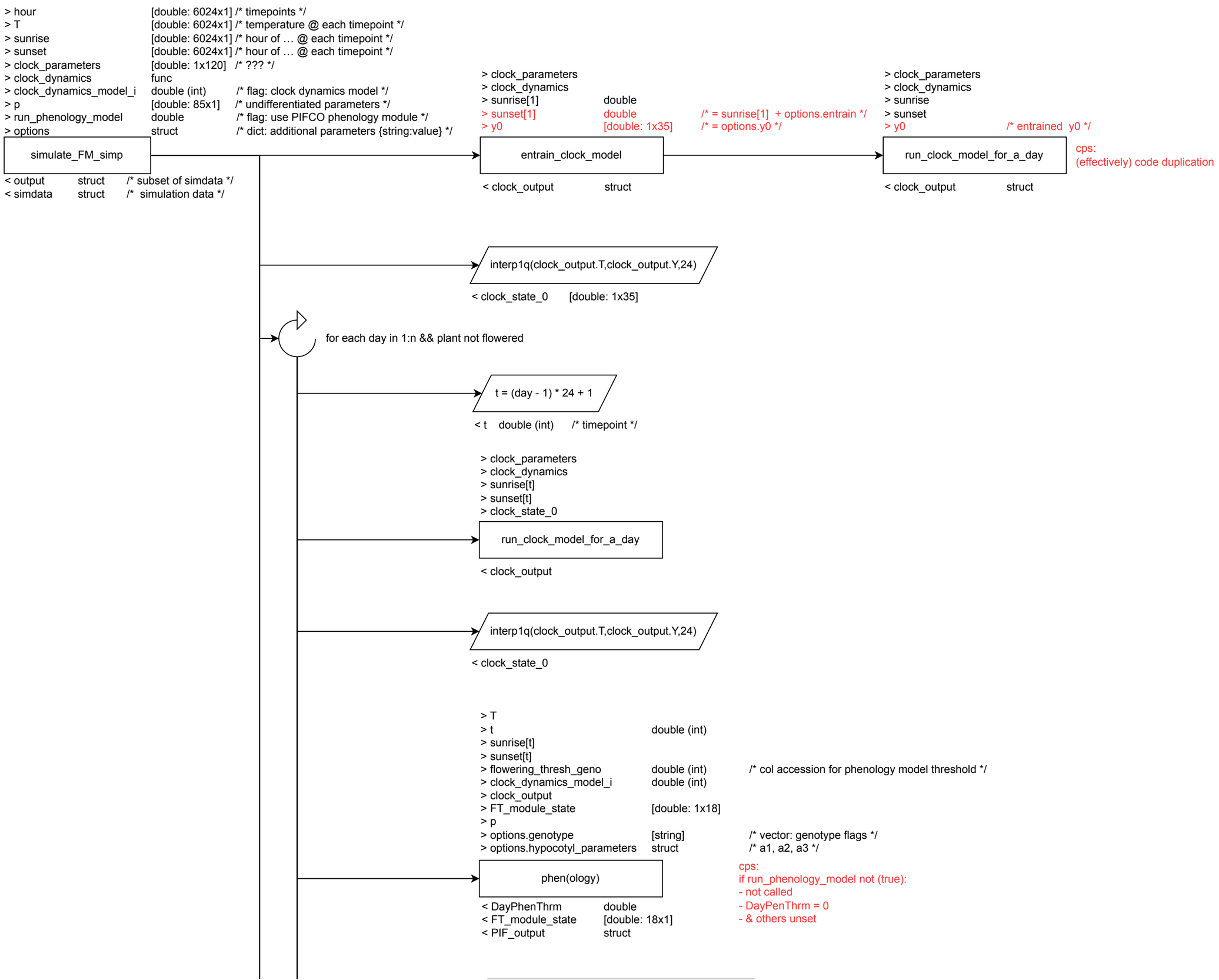

simulate_FM_simp
simulate the Framework Model v2
plot cumulative phen thrm 1:90
costfunction
plot


fit_model_to_hypocotyl_data
???
init default options
init light conditions
init model parameters
select clock parameters
< clock_parameters [double: 1x120]
@ clock_dynamics_model_i == 2
load_P2011_parameters({'wt'})
@ clock_dynamics_model_i == 4
load_F2014_parameters({'wt'})
cps:
if clock_dynamics_model_i neither?
- should align with simulate_PIF_CO_FT?
select clock model dynamics
< clock_dynamics
@ clock_dynamics_model_i == 2
wrap_P2011_model_dynamics
@ clock_dynamics_model_i == 4
wrap_F2014_model_dynamics
select initial conditions for clock
< options.y0
@ clock_dynamics_model_i == 2
wrap_P2011_model_dynamics
@ clock_dynamics_model_i == 4
wrap_F2014_model_dynamics
for photoperiod in [0, 8, 16]
update options
update env
> hour [double: 6024x1] /* timepoints */
> T [double: 6024x1] /* temperature @ each timepoint */
> sunrise [double: 6024x1] /* hour of … @ each timepoint */
> sunset [double: 6024x1] /* hour of … @ each timepoint */
> clock_parameters [double: 1x120] /* ??? */
> clock_dynamics func
> clock_dynamics_model_i double (int) /* flag: clock dynamics model */
> p [double: 85x1] /* undifferentiated parameters */
> run_phenology_model double /* flag: use PIFCO phenology module */
> options struct /* dict: additional parameters {string:value} */
simulate_FM_simp


simulate_FM_simp
simulate the Framework Model v2
costfunction
plot

Entry Points (4 / 4)

Model Core


> hour [double: 6024x1] /* timepoints */
> T [double: 6024x1] /* temperature @ each timepoint */
> sunrise [double: 6024x1] /* hour of … @ each timepoint */
> sunset [double: 6024x1] /* hour of … @ each timepoint */
> clock_parameters [double: 1x120] /* ??? */
> clock_dynamics func
> clock_dynamics_model_i double (int) /* flag: clock dynamics model */
> p [double: 85x1] /* undifferentiated parameters */
> run_phenology_model double /* flag: use PIFCO phenology module */
> options struct /* dict: additional parameters {string:value} */
simulate_FM_simp
< output struct /* subset of simdata */
< simdata struct /* simulation data */
> clock_parameters
> clock_dynamics
> sunrise[1] double
> sunset[1] double /* = sunrise[1] + options.entrain */
> y0 [double: 1x35] /* = options.y0 */
entrain_clock_model
< clock_output struct
> clock_parameters
> clock_dynamics
> sunrise
> sunset
> y0 /* entrained y0 */
run_clock_model_for_a_day
< clock_output struct
cps:
(effectively) code duplication
interp1q(clock_output.T,clock_output.Y,24)
< clock_state_0 [double: 1x35]
for each day in 1:n && plant not flowered
t = (day - 1) * 24 + 1
< t double (int) /* timepoint */
> clock_parameters
> clock_dynamics
> sunrise[t]
> sunset[t]
> clock_state_0
run_clock_model_for_a_day
< clock_output
interp1q(clock_output.T,clock_output.Y,24)
< clock_state_0
> T
> t double (int)
> sunrise[t]
> sunset[t]
> flowering_thresh_geno double (int) /* col accession for phenology model threshold */
> clock_dynamics_model_i double (int)
> clock_output
> FT_module_state [double: 1x18]
> p
> options.genotype [string] /* vector: genotype flags */
> options.hypocotyl_parameters struct /* a1, a2, a3 */
phen(ology)
< DayPhenThrm double
< FT_module_state [double: 18x1]
< PIF_output struct
cps:
if run_phenology_model not (true):
- not called
- DayPenThrm = 0
- & others unset

CumPhenThrm = CumPhenThrm + DayPhenThrm

< CumPhenThrm double /* cumulative phenology thrm */

> CumPhenThrm
> flowering_thresh_geno double (int)
> p

flowering_threshold_test

< has_flowered double (bool) /* flag: has flowered */

CumPhenThrmAll[t] = CumPhenThrm

< CumPhenThrmAll[t] [double: nx1] /* cumulative phenology thrm tracker */

simdata.metadata.hour	[double: 6024x1] = hour
simdata.metadata.T	[double: 6024x1] = T
simdata.metadata.sunrise	[double: 6024x1] = sunrise
simdata.metadata.sunset	[double: 6024x1] = sunset
simdata.metadata.photoperiod	[double: 6024x1] = sunset - sunrise
simdata.metadata.clock_parameters	[double: 1x120] = clock_parameters
simdata.metadata.p	[double: 85x1] = p
simdata.metadata.run_phenology_model	double (bool) = run_phenology_model

simdata.clock_output	struct	= clock_output	/* last value */
simdata.PIF_output	struct	= PIF_output	/* last value */
simdata.FI	double (int)	= day - 1	
simdata.Hyp	double	= PIF_output.Hyp_length	
simdata.CumPhenThrm	[double: 90x1]	= CumPhenThrmAll	

output [???] = [simdata.clock(_output),
simdata.PIF(_output),
simdata.Hyp,
sim_data.FI]

Phenology

> T	[double: 6024x1]	/* Temperature at time points */
> t	double (int)	/* timepoint */
> sunrise	double (int)	/* sunrise @ hours */
> sunset	double (int)	/* sunset @ hours */
> geno	double (int)	/* genotype index */
> clock_dynamics_model_i	double (int)	/* flag: clock dynamics model */
> clock_output	struct	/* clock model generated time series for parameters */
> FT_module_state	[double: 18x1]	/* state vector of FT model (conc. model components @ initial time point */
> p	[double: 85x1]	/* undifferentiated parameters */
> flowering_genotype	[string]	/* vector: genotype flags */
> hypocotyl_parameters	struct	/* a1, a2, a3 */

phen(ology)

< DayPhenThrm	double
< FT_module_state	[double: 18x1]
< PIF_output	struct

set-up constants based on p

> sunrise	double (int)	/* sunrise @ hours */
> sunset	double (int)	/* sunset @ hours */
> clock_output	struct	/* clock model generated time series for parameters */
> clock_dynamics_model_i	double (int)	/* flag: clock dynamics model */
> FT_module_state	[double: 1x18]	/* state vector of FT model (conc. model components @ initial time point */
> temperatures	[double: 6024x1]	/* Temperature at time points */
> flowering_genotype	[string]	/* vector: genotype flags */
> hp	struct	/* = hypocotyl_parameters: a1, a2, a3 */

simulate_PIF_CO_FT_model

< dailyFTarea	
< FT_module_state	
< PIF_output	

select clock parameters

< clock_parameters	[double: 1x120]

@ clock_dynamics_model_i < 3
load_P2011_parameters (flowering_genotype)

@ otherwise
load_F2014_parameters (flowering_genotype)

select clock model dynamics wrapper

< func_dynamics_wrapper	

@ clock_dynamics_model_i < 3
wrap_P2011_model_dynamics

@ otherwise
wrap_F2014_model_dynamics

u = func_dynamics_wrapper(
clock_output.T,clock_output.Y,clock_parameters)

< u	struct

> flowering_genotype [string] /* vector: genotype flags */
> temperature [double: 6024x1] /* Temperature at time points */

load_PIF_CO_FT_parameters.m

cps:
tweaks parameters if temperature @ 27 °C

< parameters.PIF_CO_FT

ode15s((t,y) → PIF_CO_FT_Dynamics(t, y, parameters.PIF_CO_FT, u, c), [0 c.period], FT_module_state)

< T [double: 217x1]
< Y [double: 217x18]

PIF_CO_FT_dynamics

> t double /* time */
> y [double] /* values */
> P struct /* parameters.PIF_CO_F
> c struct /* light conditions inputs *

< dydt [double: ?x?]

> Y [double: 217x18]

FT_module_state = Y(end. :)'

< FT_module_state [double: 18x1]

> T [double: 217x1]
> Y [double: 217x18]

dailyFTArea = trapz(T,Y(:,15));

< dailyFTArea double

PIF_output.T	[double: 217x1]	= T
PIF_output.Y	[double: 217x18]	= Y
PIF_output.FTarea	double	= dailyFTArea
PIF_output.Hyp_length	double	= (calculation)

< PIF_output struct

for each hour in day

Phot[hour] = calculate photoperiod component

Ve[hour] = calculate vernalisation effectiveness

cps:
uses temperature[hour]

Vh[hour] = cumulative vernalisation hours

Vern[hour] = cumulative fraction

mptu[hour] = Vern[hour] * Phot[hour] * Thrm[hour]

DayPhenThem = sum(mptu)

T */
/

> c struct /* light conditions inputs */

light_conditions

< L double /* multiplier light condition */