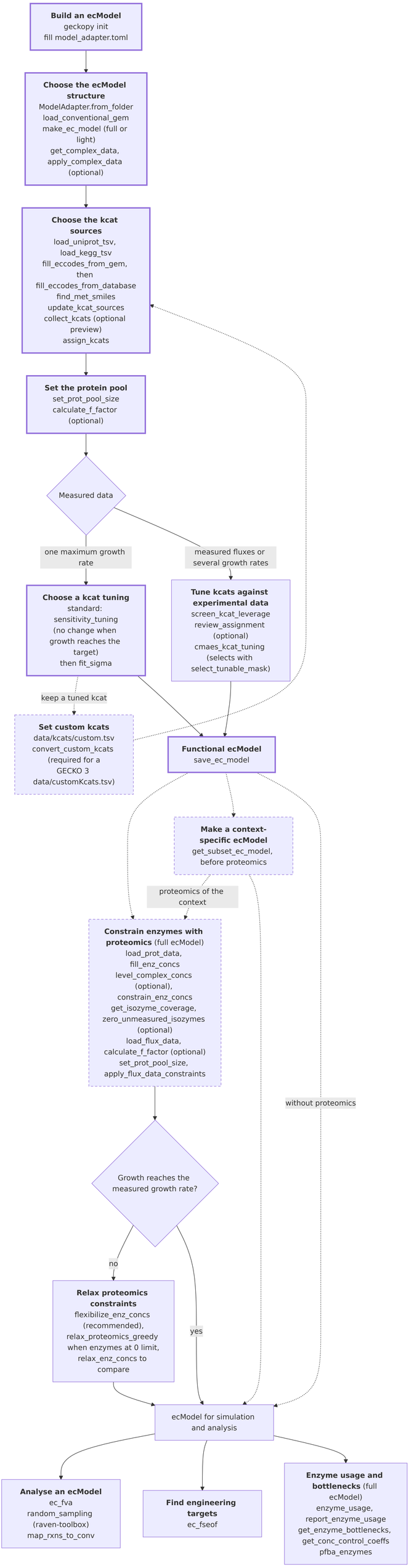

Build an ecModel
geckopy init
fill model_adapter.toml
Choose the ecModel structure
ModelAdapter.from_folder
load_conventional_gem
make_ec_model (full or light)
get_complex_data,
apply_complex_data (optional)
Choose the kcat sources
load_uniprot_tsv,
load_kegg_tsv
fill_eccodes_from_gem,
then
fill_eccodes_from_database
find_met_smiles
update_kcat_sources
collect_kcats (optional preview)
assign_kcats
Set the protein pool
set_prot_pool_size
calculate_f_factor (optional)
Measured data
one maximum growth rate
measured fluxes or several growth rates
Choose a kcat tuning
standard:
sensitivity_tuning
(no change when growth reaches the target)
then fit_sigma
Tune kcats against experimental data
screen_kcat_leverage
review_assignment (optional)
cmaes_kcat_tuning
(selects with select_tunable_mask)
keep a tuned kcat
Set custom kcats
data/kcats/custom.tsv
convert_custom_kcats
(required for a GECKO 3 data/customKcats.tsv)
Functional ecModel
save_ec_model
Make a context-specific ecModel
get_subset_ec_model,
before proteomics
proteomics of the context
Constrain enzymes with proteomics (full ecModel)
load_prot_data,
fill_enz_concs
level_complex_concs (optional),
constrain_enz_concs
get_isozyme_coverage,
zero_unmeasured_isozymes (optional)
load_flux_data,
calculate_f_factor (optional)
set_prot_pool_size,
apply_flux_data_constraints
without proteomics
Growth reaches the measured growth rate?
no
Relax proteomics constraints
flexibilize_enz_concs (recommended),
relax_proteomics_greedy
when enzymes at 0 limit,
relax_enz_concs to compare
yes
ecModel for simulation and analysis
Analyse an ecModel
ec_fva
random_sampling
(raven-toolbox)
map_rxns_to_conv
Find engineering targets
ec_fseof
Enzyme usage and bottlenecks (full ecModel)
enzyme_usage,
report_enzyme_usage
get_enzyme_bottlenecks,
get_conc_control_coeffs
pfba_enzymes