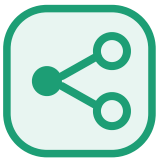


MAXIMALABS

The Process Simulation Handbook

A working introduction to process simulation — thermodynamics, flash, reactors, separations, heat integration, fluid handling, solids operations, dynamics, and analysis — taught entirely against a real, running simulator.

2026-08-17

Contents

CHAPTER 1

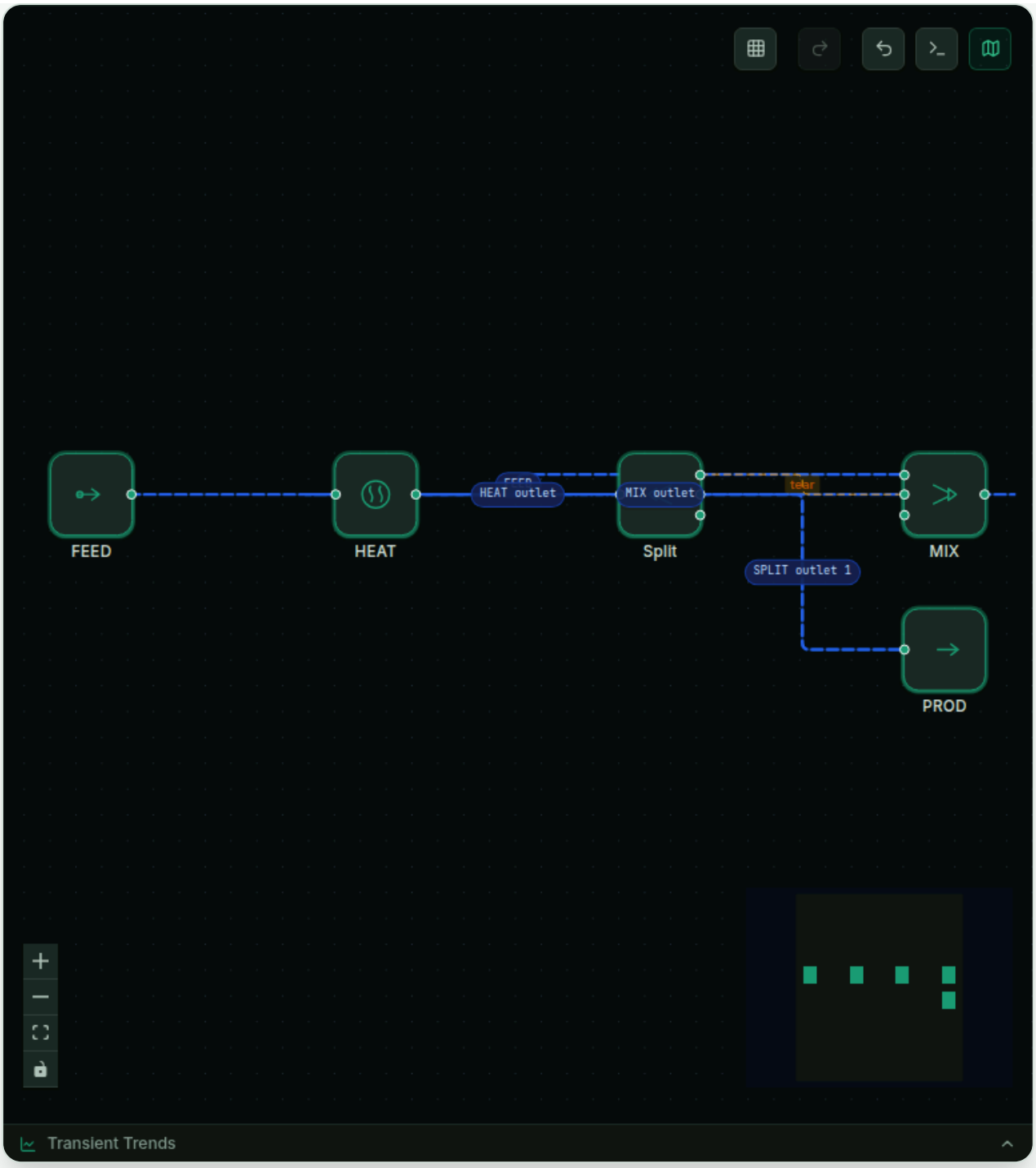
Simulation Fundamentals

Before any thermodynamics or unit-operation detail, a process simulator has to answer one question: *given a set of connected equipment and some specifications, what are all the stream flows, temperatures, pressures, and compositions?* This chapter is the machinery that answers it — the same machinery under Aspen Plus and HYSYS, but in MaximaLabs it's not hidden. You can watch it work.

1.1 A flowsheet is a graph

A steady-state model is a **directed graph**: nodes are unit operations (feeds, reactors, columns, exchangers, products) and edges are the **streams** that carry material between them. Every stream is the same object everywhere in MaximaLabs — flow, temperature, pressure, composition, phase, enthalpy, entropy — so the simulator, the cost estimator, and the reports all read one representation.

The example below is a minimal loop: a feed mixes with a recycle, is heated, and split — part leaves as product, part returns. Notice the orange **tear** badge on the returning stream; we'll come back to why it's marked (§1.4).



The reference 'Recycle loop' example on the MaximaLabs canvas. The returning stream is flagged as the tear (orange badge) — the edge the solver cuts to break the cycle.

1.2 Two ways to solve it: sequential-modular vs. equation-oriented

There are two classical strategies for solving that graph, and MaximaLabs implements both — you pick from the **Steady-state** strategy menu in the status bar.

Sequential-modular (SM). Each unit is a self-contained module: given its inlets and parameters, it computes its outlets. The solver visits units in topological order,

passing streams forward. It's robust and easy to reason about — but a recycle creates a cycle, so there's no "first" unit; SM has to guess a stream, iterate, and converge (§1.4–1.5).

Equation-oriented (EO). Instead of marching unit by unit, EO collects *every* model equation for the whole flowsheet into one large system and solves it *simultaneously* with a Newton method:

$$F(x) = 0, \qquad x_{k+1} = x_k - J(x_k)^{-1} F(x_k), \quad J = \frac{\partial F}{\partial x}$$

x	The stacked vector of every unknown stream variable in the whole flowsheet (flows, temperatures, pressures, compositions) — everything the solver has to find at once.
$F(x)$	The stacked vector of every model equation (material balances, energy balances, equilibrium relations, summation constraints) evaluated at the current guess x — zero exactly when x is the converged solution.
J	The Jacobian — the matrix of every equation's sensitivity to every unknown ($\partial F_i / \partial x_j$), rebuilt (or approximated) at each Newton step.
$x_k,\ x_{k+1}$	The unknown vector at iteration k and the next, improved iteration — Newton's method repeats this update until F(x) is below tolerance.

Here

$$x$$

stacks all the unknown stream variables and

$$F$$

stacks all the material balances, energy balances, equilibrium and summation relations. A recycle is no longer special — it's just a few more equations in the same system. This is the approach AVEVA SimCentral, gPROMS, and IDAES use, and it's why tightly-coupled recycles that crawl in SM converge cleanly in EO.

WORKED EXAMPLE — THE AMMONIA LOOP'S EO SYSTEM (§1.6)

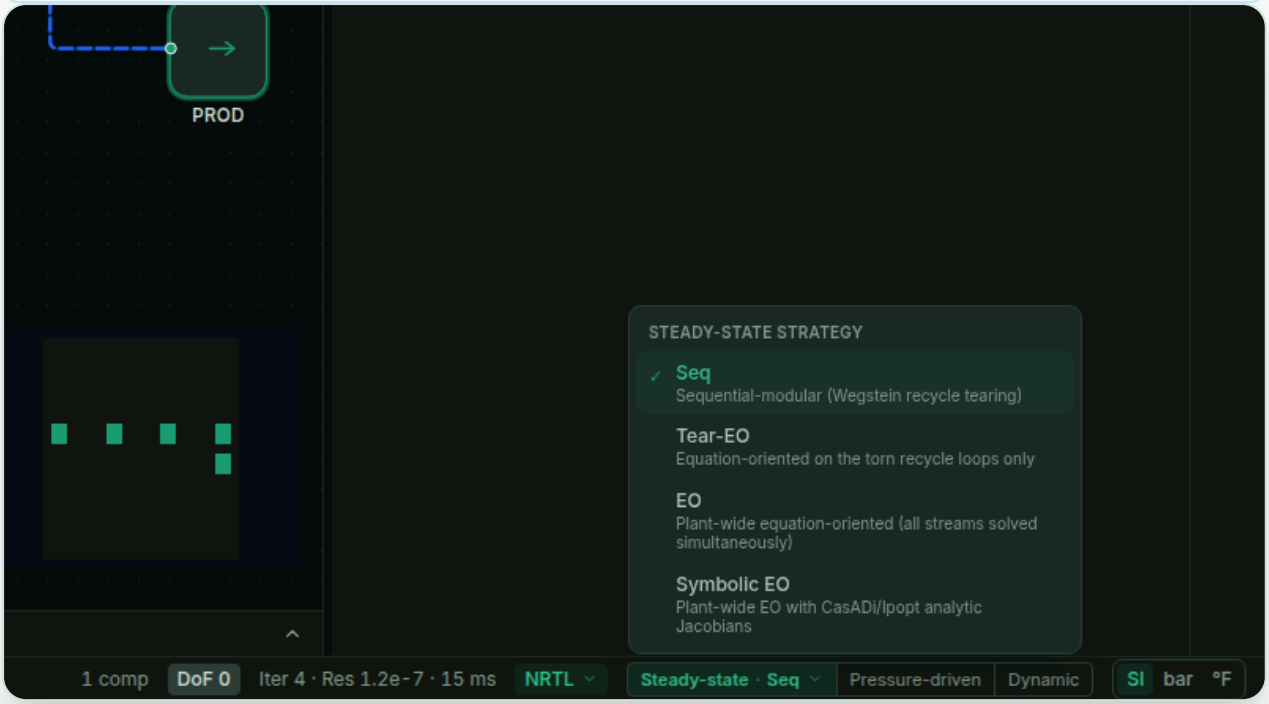
For the high-recycle ammonia loop below, MaximaLabs' plant-wide EO mode stacks

$$x$$

and

$$F(x)$$

into a system with **28 unknowns and 28 residual equations** (the same square system shown in §1.3's Solver-math screenshot) and solves it in **one** simultaneous Newton solve — versus the ~78 tear passes the sequential-modular strategy needs for the identical flowsheet (§1.5–1.6).



The strategy menu: Sequential-modular, Tear-EO (EO only on the torn loops), plant-wide EO, and Symbolic EO (CasADi/Ipopt analytic Jacobians). The status bar carries the live DoF pill and iteration/residual readout.

1.3 Degrees of freedom

A model is solvable only if it's **square** — as many independent equations as unknowns. The degrees of freedom are

$$N_{\text{DOF}} = N_{\text{variables}} - N_{\text{equations}}$$

$N_{\text{variables}}$	Total unknown stream/unit variables in the flowsheet (every flow, temperature, pressure, and composition not yet fixed by a spec).
$N_{\text{equations}}$	Total independent model equations (material balances, energy balances, equilibrium/summation relations) contributed by every unit op.

N_{DOF}	Degrees of freedom — the count of variables left over once the equations are subtracted. Zero means the model is exactly solvable ("square").

If

$$N_{\mathrm{DOF}} > 0$$

the flowsheet is *under-specified* (you must fix more variables); if

$$N_{\mathrm{DOF}} < 0$$

it's *over-specified* (a spec has to give). MaximaLabs computes this structurally from the graph and shows it live as the **DoF** pill in the status bar — so you see under/over-specification *before* you ever hit a cryptic solver failure. The Solver-math panel makes it concrete: the equation-oriented system for the recycle example below is exactly square (28 unknowns, 28 residual equations, DoF 0), and every residual is labelled by the unit and the balance it enforces.

WORKED EXAMPLE — THE RECYCLE-LOOP'S DOF COUNT

The Solver-math ▸ Structure screenshot below reads it directly:

$$N_{\mathrm{variables}} = 28$$

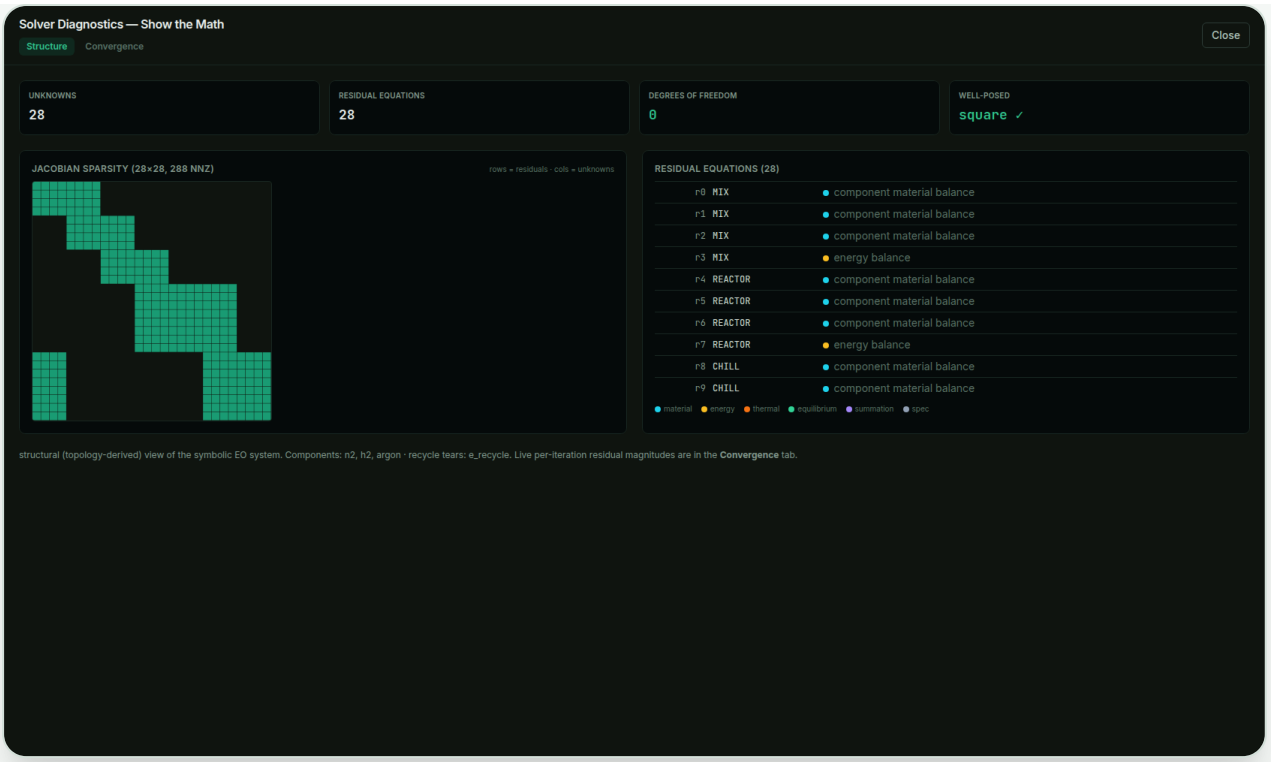
,

$$N_{\mathrm{equations}} = 28$$

, so

$$N_{\mathrm{DOF}} = 28 - 28 = 0$$

— exactly square, well-posed, no missing or redundant specs. That's the number that lets the solver even attempt a solve.



Solver-math ▸ Structure: the whole equation-oriented system laid bare — 28 unknowns = 28 equations ⇒ DoF 0 (square, well-posed), the Jacobian sparsity pattern, and each residual tagged material/energy/equilibrium/summation. Aspen solves the same system; here you can see it.

1.4 Recycles and tear streams

A recycle makes the graph **cyclic**, so sequential-modular can't just march forward — computing the mixer needs the recycle stream, but computing the recycle stream needs everything downstream of the mixer. The classic fix is **tearing**: cut one edge in the cycle (the *tear stream*), guess its values, march around the loop, and compare the value that comes back with the guess. The mismatch is the tear residual:

$$r = x_{\text{calc}} - x_{\text{guess}} \;\to\; 0$$

x_{guess}	The tear stream's assumed values (flow, T, P, composition) at the start of a pass — the guess that lets sequential-modular execution start despite the cycle.
x_{calc}	What that same stream computes out to after marching all the way around the loop once with that guess.
r	The tear residual — the gap between what was guessed and what came back. The solver keeps passing until r is below tolerance.

MaximaLabs picks the tear automatically and marks it on the canvas (the orange badge you saw in §1.1), then iterates until the residual is below tolerance. You can also pin the tear's initial guess yourself for a hard-to-start loop.

WORKED EXAMPLE — THE TEAR RESIDUAL, AMMONIA LOOP

§1.5–1.6's high-recycle ammonia loop starts its first tear pass with residual

$$r \approx 7 \times 10^{-2}$$

(the initial guess is far from the real recycle composition) and Wegstein-accelerated passes drive it down to

$$r \approx 9.01 \times 10^{-7}$$

after ~78 passes — the point at which MaximaLabs calls the loop converged.

1.5 Convergence: direct substitution and Wegstein

Marching around a torn loop is a fixed-point iteration

$$x_{k+1} = g(x_k)$$

, where

$$g$$

is "go around the loop once." The simplest scheme, **direct substitution**, just feeds the result back in — but for a tightly-coupled loop it can crawl or oscillate. **Wegstein acceleration** fixes that by extrapolating each component from the last two passes:

$$s = \frac{g(x_k) - g(x_{k-1})}{x_k - x_{k-1}}, \qquad q = \frac{s}{s - 1}$$

$$x_{k+1} = q\, x_k + (1 - q)\, g(x_k)$$

$g(x_k)$	One trip around the torn loop starting from the current guess x_k — the same "march the loop once" operation direct substitution uses on its own.

s	The observed local slope of g between the last two passes — how sensitive the loop's output is to its input, estimated numerically rather than assumed.
q	The Wegstein extrapolation factor built from s — near 0 behaves like plain direct substitution, negative q overshoots deliberately to escape a slow crawl.
x_{k+1}	The next guess, blended between the raw substitution result g(x_k) and the current x_k by q — this is what replaces plain direct substitution.

The high-recycle ammonia loop below has a recycle-to-fresh ratio of about 5:1 — a genuinely stiff loop. In sequential-modular mode MaximaLabs tears it and takes ~78 Wegstein passes to drive the residual from

$$7 \times 10^{-2}$$

down to

$$9 \times 10^{-7}$$

. The Solver-math ▸ Convergence tab plots that trajectory on a log scale, so "did it converge, and how fast" is something you *see*, not guess:

WORKED EXAMPLE — WEGSTEIN ON THE AMMONIA LOOP

Same loop as above:

$$78$$

Wegstein-accelerated passes take the scaled tear residual from

$$\approx 7 \times 10^{-2}$$

to a final

$$9.01 \times 10^{-7}$$

. Plain direct substitution (

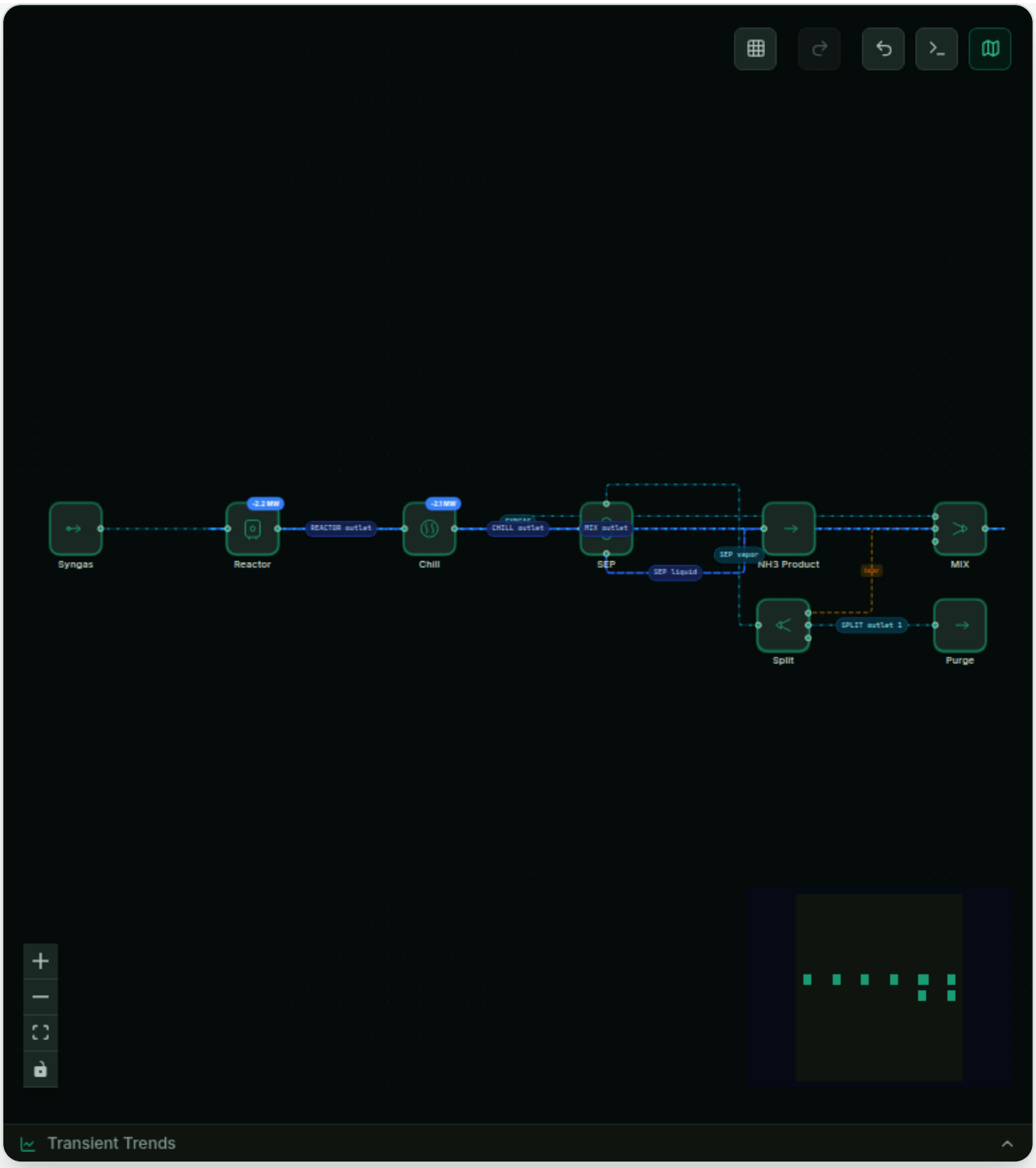
$$q = 0$$

always) would need many more passes on a loop this stiff — extrapolating with the observed slope s is what buys the speed-up.

Solver-math ▸ Convergence for the ammonia loop in sequential-modular mode: 78 Wegstein iterations, residual driven to 9.01×10^{-7}. The same loop in EO mode closes all ~35 streams in one Newton solve — same answer, different path.

1.6 Worked example: solve one loop two ways

Put it together on the high-recycle ammonia synthesis loop — fresh syngas (N_2 + $3H_2$, with argon inert) mixes with a large recycle, reacts ~10% per pass, chills so ammonia condenses out as product, and the unreacted gas recycles, with a small purge to bleed the argon. It's the canonical case where solution strategy actually matters.



The high-recycle ammonia loop example, converged, on the canvas.

REPRODUCE IT IN YOUR BROWSER

1. Open the example directly: high-recycle ammonia loop (or use the app's example gallery).
2. Leave the strategy on **Steady-state · Seq** (the status-bar toggle) and click **Run simulation**. Watch the iteration/residual readout tick — this loop takes tens of tear passes.
3. Open **Solver math** (right rail ▸ Analysis & reports) and switch to the **Convergence** tab to see the Wegstein residual trajectory, then **Structure** to see the square DoF-0

system and Jacobian.

4. Now change the strategy to **EO** (Steady-state ▾ ▸ EO) and **Run** again. It reaches the *same* converged answer in a single simultaneous Newton solve instead of ~78 tear passes — the point of the chapter is the *path*, not the result.
5. Read the converged stream table (right panel) and confirm the mass balance closes around the loop, argon and all.

That's the whole fundamentals loop: build a graph, check it's square, tear and converge the recycles (or solve everything at once), and read the answer. Every later chapter — thermo, reactors, distillation — is a richer *node* inside this same machine.

1.7 Exercises

PRACTICE

Work each problem yourself first, then reveal the solution to check it. Where a problem says so, reproduce it live in MaximaLabs — the solver is the answer key.

1. WARM-UP

   A flowsheet runs **feed → mixer → reactor → separator**, and the separator sends a product out while recycling its other outlet back to the mixer. How many streams must the sequential-modular solver **tear** to compute the loop, and which one?

   Show solution

2. CORE

   You draw **feed → heater → reactor** with the feed fully specified, and the degrees-of-freedom indicator reads **−1 (under-specified)**. What single spec is missing, and what are your two choices for it?

   Show solution

3. CHALLENGE

   A recycle carries five units of flow back for every one unit of fresh feed (recycle ratio ≈ 5). Explain why **sequential-modular** needs many tear

passes to converge this loop while **equation-oriented** solves it in a single Newton solve — then watch it in the high-recycle ammonia example.

Show solution

CHAPTER 2

Thermodynamic & Physical Property Models

Chapter 1 solved the flowsheet as a graph. But every unit in that graph — every flash, column stage, and reactor outlet — is only as correct as the **property model** behind it. Choosing that model is the single most consequential decision in a simulation, and the one most likely to be wrong. This chapter is how you choose, and how MaximaLabs helps you get it right.

2.1 The most important choice you make

The same flowsheet, solved with two different property methods, can give two completely different answers — not slightly different, *qualitatively* different. The textbook case is ethanol–water: it forms an **azeotrope** at about 89 mol% ethanol, a hard ceiling that ordinary distillation cannot cross. A model that captures the strong non-ideality of the liquid (an activity-coefficient model) predicts that azeotrope; a bare cubic equation of state, tuned for hydrocarbons, does not — so it will happily (and wrongly) report a column reaching 99.9% ethanol overhead. Same column, same specs; the thermodynamics decide whether the answer is physical.

2.2 Equations of state

An equation of state (EoS) relates pressure, volume, and temperature for both phases from one set of parameters. The workhorses are the cubics — Peng-Robinson (PR) and Soave-Redlich-Kwong (SRK):

$$P = \frac{RT}{v - b} - \frac{a\,\alpha(T)}{v^2 + 2bv - b^2} \qquad \text{(Peng-Robinson)}$$

$P,\ T,\ v$	Pressure, temperature, and molar volume — the three state variables the EoS relates.
R	Universal gas constant.

$a,\ b$	Peng-Robinson pure-component energy and co-volume parameters, computed once from each component's critical temperature/pressure and acentric factor.
$\alpha(T)$	A temperature-dependent correction factor on a, fit so the EoS reproduces the pure component's own vapor-pressure curve near its critical point.

The EoS gives fugacity coefficients

$$\hat{\varphi}_i$$

in each phase, and phase equilibrium is where they match, which fixes the K-value

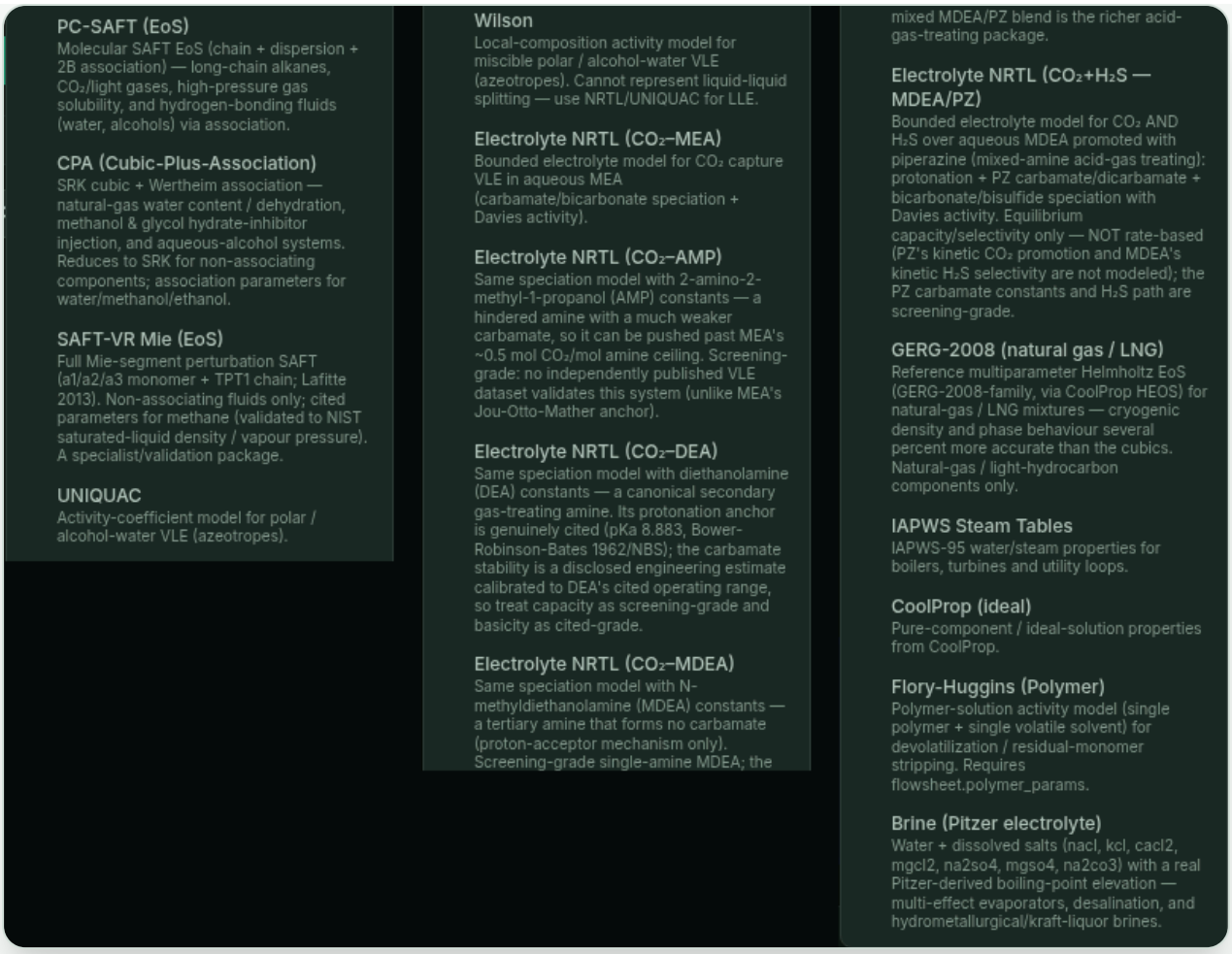
$$K_i = y_i / x_i$$

:

$$\hat{\varphi}_i^{\,V}\, y_i = \hat{\varphi}_i^{\,L}\, x_i \quad \Rightarrow \quad K_i = \frac{\hat{\varphi}_i^{\,L}}{\hat{\varphi}_i^{\,V}}$$

$\hat{\varphi}_i^{\,V},\ \hat{\varphi}_i^{\,L}$	Fugacity coefficient of component i in the vapor and liquid mixture — both computed from the same EoS at the mixture's T, P, and composition, so no separate liquid-phase model is needed.
$y_i,\ x_i$	Vapor- and liquid-phase mole fraction of component i.
K_i	Equilibrium ratio K_i = y_i/x_i, fixed once the fugacities of both phases match.

Cubics are the right default for non-polar, hydrocarbon, and high-pressure gas systems — natural gas, refinery cuts, compressors. MaximaLabs ships PR and SRK plus more specialized equations of state (PC-SAFT, CPA for associating fluids, GERG-2008 for LNG). You pick the property method from the status bar; the menu spans equations of state, activity models, electrolytes, steam, and polymer models:



The property-method menu (status-bar thermo pill), scrolled to show the breadth beyond the cubics — SAFT-family equations of state, activity models (UNIQUAC, Wilson), electrolytes, steam tables, and a polymer model — each with what it's for. Peng-Robinson, SRK, and NRTL sit at the top of the same list; switching any of them re-solves the flowsheet.

2.3 Activity-coefficient models

For a strongly non-ideal *liquid* — alcohols, water, acids, anything with hydrogen bonding — the right tool is an activity-coefficient model. Here the liquid non-ideality lives in an activity coefficient

$$\gamma_i$$

, and VLE is the modified Raoult's law:

$$y_i \, \varphi_i^{V} \, P = x_i \, \gamma_i \, P_i^{\mathrm{sat}}(T)$$

$y_i,\ x_i$	Vapor- and liquid-phase mole fraction of component i.

φ_i^V	Vapor-phase fugacity coefficient of i (≈1 near atmospheric pressure, where this method is normally used).
P	System (total) pressure.
γ_i	Liquid activity coefficient of i — the non-ideality correction the activity model below supplies; 1 is the ideal (Raoult's-law) limit.
$P_i^{\text{sat}}(T)$	Pure-component vapor pressure of i at the mixture temperature T.

The default in MaximaLabs for polar systems is **NRTL** (Non-Random Two-Liquid), whose activity coefficient for component

$$i$$

is

$$\ln \gamma_i = \frac{\sum_j x_j \tau_{ji} G_{ji}}{\sum_k x_k G_{ki}} + \sum_j \frac{x_j G_{ij}}{\sum_k x_k G_{kj}} \left(\tau_{ij} - \frac{\sum_m x_m \tau_{mj} G_{mj}}{\sum_k x_k G_{kj}} \right)$$

$$G_{ij} = \exp(-\alpha_{ij} \tau_{ij}), \qquad \tau_{ij} = a_{ij} + b_{ij}/T$$

γ_i	Activity coefficient of component i being solved for.
$x_i,\ x_j,\ x_k,\ x_m$	Liquid mole fractions — j, k, m range over every component (the sums); i is the one component gamma is evaluated for.
τ_{ij}	Dimensionless NRTL interaction energy for the ordered pair i,j (asymmetric — tau_ij is not tau_ji).
G_{ij}	NRTL local-composition weighting factor for the pair i,j — how strongly j clusters around i relative to i itself.
α_{ij}	Non-randomness parameter for the pair (a third fitted constant, often held near 0.2-0.3 across many families rather than regressed).
$a_{ij},\ b_{ij}$	The two fitted binary interaction constants — a is temperature-independent, b/T carries the temperature dependence.
T	Temperature (K).

The binary interaction parameters

$$a_{ij}, b_{ij}, \alpha_{ij}$$

are the data dependency — MaximaLabs carries a DECHEMA/databank set (and you can regress or override your own).

WORKED EXAMPLE — NRTL ACTIVITY COEFFICIENTS FOR ETHANOL-WATER

MaximaLabs' pinned ethanol(1)-water(2) NRTL parameters (regressed to DECHEMA VLE data):

$$a_{12} = -0.8009,\ b_{12} = 246.18,\ a_{21} = 3.4578,\ b_{21} = -586.08,\ \alpha = 0.30$$

. At

$$x = (0.4,\ 0.6)$$

,

$$T = 358\ \mathrm{K}$$

(the same feed as Chapter 3's flash example):

$$\tau_{12} = -0.113, \qquad \tau_{21} = 1.821$$

$$G_{12} = \exp(-0.30 \times (-0.113)) = 1.035, \qquad G_{21} = \exp(-0.30 \times 1.821) = 0.579$$

$$\gamma_{\mathrm{ethanol}} = 1.423, \qquad \gamma_{\mathrm{water}} = 1.327$$

Both exceed 1 — a positive deviation from Raoult's law, the hydrogen-bonding non-ideality that eventually produces the ethanol-water azeotrope (§2.5). A cubic EoS with no activity model built in cannot generate this.

2.4 The rest of the activity-model family

NRTL isn't the only local-composition model, and it isn't predictive — every one of these needs fitted binary parameters. Four more are worth knowing by their actual equations, not just their names, because each makes a different trade between simplicity, physical grounding, and whether it needs data at all.

Van Laar is the oldest of the family — built directly on the van der Waals equation for regular solutions, two parameters, no temperature dependence. For a binary:

$$\frac{G^{ex}}{x_1 x_2 RT} = \frac{A_{12}A_{21}}{A_{12}x_1 + A_{21}x_2}$$

G^{ex}	Excess Gibbs energy of mixing (J/mol) — the departure from an ideal solution.
$x_1,\ x_2$	Liquid mole fractions of components 1 and 2.
$R,\ T$	Gas constant and temperature (van Laar's constants carry no further T dependence themselves).
$A_{12},\ A_{21}$	The two fitted, temperature-independent van Laar binary constants.

which splits into the two per-component activity coefficients MaximaLabs actually evaluates:

$$\ln \gamma_1 = A_{12}\left(\frac{A_{21}x_2}{A_{12}x_1 + A_{21}x_2}\right)^2, \qquad \ln \gamma_2 = A_{21}\left(\frac{A_{12}x_1}{A_{12}x_1 + A_{21}x_2}\right)^2$$

$\gamma_1,\ \gamma_2$	Activity coefficients of components 1 and 2.
$A_{12},\ A_{21}$	Same two fitted constants as above — note the asymmetry: A_12 weights gamma_1 with A_21, and vice versa.
$x_1,\ x_2$	Liquid mole fractions.

Its simplicity is also its limit — no temperature dependence means it can't track VLE over a wide range, but it's cheap to fit and still shows up for quick screening. It's a real, selectable MaximaLabs thermo package (

`van-laar`

, regressed to the same cited DECHEMA ethanol-water points as NRTL/UNIQUAC/Wilson) — not just the textbook equation above, though its 2-constant fit is honestly a touch looser (it places the azeotrope at ~91.6 mol% vs. the 89.4% reference, a known trait of having no temperature dependence).

WORKED EXAMPLE — VAN LAAR ACTIVITY COEFFICIENTS FOR ETHANOL-WATER

MaximaLabs' pinned ethanol(1)-water(2) van Laar constants:

$$A_{12} = 1.7117,\ A_{21} = 0.9101$$

. At the same

$$x = (0.4,\ 0.6)$$

,

$$T = 358\ \mathrm{K}$$

feed as the NRTL example above (van Laar ignores T, so this holds at any temperature):

$$\ln\gamma_1 = 0.337 \Rightarrow \gamma_{\text{ethanol}} = 1.401, \qquad \ln\gamma_2 = 0.282 \Rightarrow \gamma_{\text{water}} = 1.325$$

Close to NRTL's 1.423 / 1.327 at this one composition — the two-constant fit's known weakness only shows up once you sweep across the full composition range and compare azeotrope location, not at a single point.

Wilson introduced *local composition* — the idea that the composition immediately around a molecule differs from the bulk because of energetic preference — which every model after it (NRTL, UNIQUAC) inherits:

$$\ln\gamma_1 = -\ln(x_1 + \Lambda_{12}x_2) + x_2\left(\frac{\Lambda_{12}}{x_1 + \Lambda_{12}x_2} - \frac{\Lambda_{21}}{x_2 + \Lambda_{21}x_1}\right)$$

γ_1	Activity coefficient of component 1 (ethanol here); gamma_2 follows the same form with subscripts 1 and 2 swapped.
$x_1,\ x_2$	Liquid mole fractions.
$\Lambda_{12},\ \Lambda_{21}$	Wilson local-composition parameters, Lambda_ij = (v_j/v_i)·exp(-a_ij/T), from the pure-liquid molar volumes v_i and two fitted binary energies a_ij.

Wilson fits VLE well with only two parameters, but structurally cannot predict liquid-liquid splitting — for that you need NRTL or UNIQUAC.

WORKED EXAMPLE — WILSON ACTIVITY COEFFICIENTS FOR ETHANOL-WATER

MaximaLabs' pinned molar volumes

$$v_{\text{ethanol}} = 58.68,\ v_{\text{water}} = 18.07\ \mathrm{cm}^3/\mathrm{mol}$$

and energy parameters

$$a_{12} = 193.50\ \mathrm{K},\ a_{21} = 472.53\ \mathrm{K}$$

. At

$$x = (0.4,\ 0.6)$$

,

$$T = 358\ \mathrm{K}$$

:

$$\Lambda_{12} = 0.179, \qquad \Lambda_{21} = 0.868$$

$$\gamma_{\mathrm{ethanol}} = 1.405, \qquad \gamma_{\mathrm{water}} = 1.323$$

Again close to NRTL/Van Laar at this composition — all four local-composition models agree reasonably well near the middle of the range; they diverge more sharply near the azeotrope, which is exactly why MaximaLabs validates each against the same published data rather than trusting any one model's equation alone.

UNIQUAC (UNIversal QUAsi-Chemical) splits the excess Gibbs energy into a *combinatorial* term (molecule size/shape, from pure-component data alone) and a *residual* term (the energetic interactions, from two fitted parameters per pair):

$$\frac{G^{ex}_{comb}}{RT} = x_1 \ln \frac{\phi_1}{x_1} + x_2 \ln \frac{\phi_2}{x_2} + \frac{z}{2}\left(q_1 x_1 \ln \frac{\theta_1}{\phi_1} + q_2 x_2 \ln \frac{\theta_2}{\phi_2} \right)$$

G^{ex}_{comb}	Combinatorial part of the excess Gibbs energy — accounts for molecule size/shape differences alone, from pure-component data, independent of energetics.
$x_1,\ x_2$	Liquid mole fractions.
$\phi_1,\ \phi_2$	Segment (volume) fractions, from r_i below.
$\theta_1,\ \theta_2$	Surface-area fractions, from q_i below.
$q_1,\ q_2$	Pure-component surface-area parameters.
z	Lattice coordination number, fixed at 10 in MaximaLabs' implementation.

$$\frac{G^{ex}_{res}}{RT} = -q_1 x_1 \ln(\theta_1 + \theta_2 \tau_{21}) - q_2 x_2 \ln(\theta_2 + \theta_1 \tau_{12})$$

G^{ex}_{res}	Residual part of the excess Gibbs energy — the energetic contribution, carrying the two fitted binary parameters.
$\tau_{12},\ \tau_{21}$	UNIQUAC binary interaction parameters (below), asymmetric between the two directions.

$\theta_1,\ \theta_2$	Surface-area fractions, same as above.
$q_1,\ q_2$	Pure-component surface-area parameters.

where the segment fraction

$$\phi_i = x_i r_i / (x_1 r_1 + x_2 r_2)$$

and surface-area fraction

$$\theta_i = x_i q_i / (x_1 q_1 + x_2 q_2)$$

come from pure-component size (

$$r_i$$

) and surface-area (

$$q_i$$

) parameters, and

$$\tau_{ij} = \exp(-(u_{ji} - u_{ii})/RT)$$

carries the two fitted binary energies. The coordination number

$$z = 10$$

.

WORKED EXAMPLE — UNIQUAC ACTIVITY COEFFICIENTS FOR ETHANOL-WATER

MaximaLabs' pinned van der Waals parameters

$$r_{\text{ethanol}} = 2.1055,\ q_{\text{ethanol}} = 1.9720,\ r_{\text{water}} = 0.9200,\ q_{\text{water}} = 1.4000$$

and interaction parameters

$$a_{12} = 19.40\ \text{K},\ a_{21} = 125.36\ \text{K}$$

(in

$$\tau_{ij} = \exp(-a_{ij}/T)$$

). At

$$x = (0.4,\ 0.6)$$

,

$$T = 358\ \mathrm{K}$$

:

$$\phi_1 = 0.604, \quad \phi_2 = 0.396, \qquad \theta_1 = 0.484, \quad \theta_2 = 0.516$$

$$\tau_{12} = 0.947, \qquad \tau_{21} = 0.705$$

$$\gamma_{\mathrm{ethanol}} = 1.411, \qquad \gamma_{\mathrm{water}} = 1.324$$

A fourth independent confirmation, within a few percent of NRTL/Van Laar/Wilson at this composition — the four models are built on different assumptions but agree closely away from the azeotrope, which is exactly the region where the choice of model matters least.

UNIFAC (UNIversal Functional Activity Coefficient) is the *predictive* answer to "what if I have no binary data at all": it reuses UNIQUAC's exact combinatorial/residual form, but builds

$$r_i$$

,

$$q_i$$

, and the interaction terms from the molecule's *functional groups* rather than the whole molecule —

$$r_i = \sum_k \nu_k^{(i)} R_k, \qquad q_i = \sum_k \nu_k^{(i)} Q_k$$

$r_i,\ q_i$	Molecule i's overall volume and surface-area parameters — the same r_i, q_i UNIQUAC's equations use, now built up rather than looked up.
$\nu_k^{(i)}$	Number of group k that molecule i contains (an integer count from its structure — e.g. ethanol has one -CH3, one -CH2-, one -OH).

$R_k,\ Q_k$	Tabulated volume/area parameters for functional group k, from the UNIFAC group table — not fitted per molecule or per pair.

$$\ln \gamma_i^{res} = \sum_k \nu_k^{(i)} \left(\ln \Gamma_k - \ln \Gamma_k^{(i)} \right)$$

γ_i^{res}	Residual (energetic) part of component i's activity coefficient.
$\nu_k^{(i)}$	Count of group k in molecule i, same as above.
Γ_k	Group k's activity coefficient evaluated in the actual mixture (all groups from every component, pooled).
$\Gamma_k^{(i)}$	Group k's activity coefficient evaluated in a reference solution of pure component i — subtracting this out is what makes the term vanish for a molecule made of only one group type.

where

$$\nu_k^{(i)}$$

counts how many of group

$$k$$

molecule

$$i$$

contains,

$$R_k, Q_k$$

are tabulated group volume/area parameters,

$$\Gamma_k$$

is group

$$k$$

's activity coefficient in the actual mixture, and

$$\Gamma_k^{(i)}$$

is the same group's coefficient in a reference mixture of pure

$$i$$

. Break ethanol into -CH_3, -CH_2-, and -OH groups, water into H_2O, look up the tabulated group interaction energies, and UNIFAC predicts the γ's — no fitted binary data for *this specific pair* required. It's the fallback MaximaLabs' Property Estimator (§2.6) reaches for when no measured VLE exists for a new mixture; the trade is accuracy — a regressed NRTL/UNIQUAC fit to real data beats a UNIFAC prediction whenever that data exists.

WORKED EXAMPLE — UNIFAC PREDICTION VS. THE REGRESSED NRTL FIT

Run MaximaLabs' original-UNIFAC group decomposition on the same ethanol-water feed,

$$x = (0.4,\ 0.6)$$

,

$$T = 358\ \mathrm{K}$$

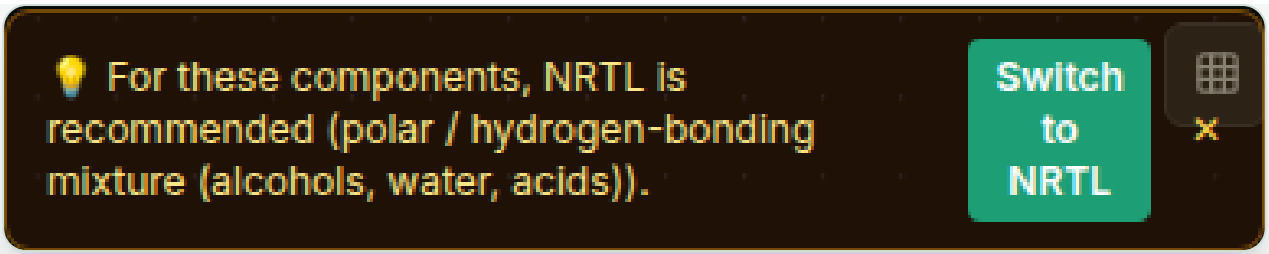
, with *no ethanol-water-specific fitted parameter at all* — only the generic -CH2-/-OH/H2O group interactions:

$$\gamma_{\mathrm{ethanol}} = 1.393, \qquad \gamma_{\mathrm{water}} = 1.345$$

Compare that to NRTL's regressed 1.423 / 1.327 above — UNIFAC lands within about 2% having seen zero data for this specific pair, which is why it's the property estimator's fallback for an uncataloged mixture rather than a guess.

2.5 Choosing a property method

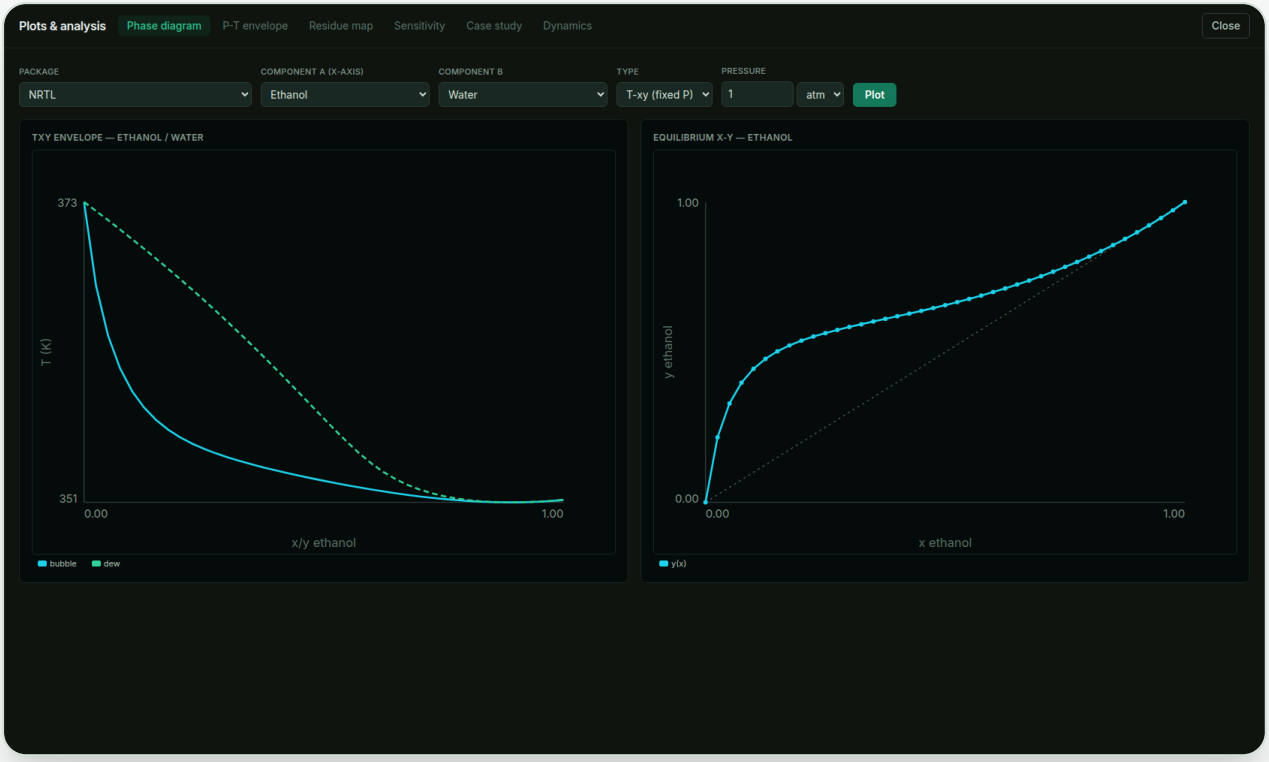
The classic decision reduces to a short tree: polar / hydrogen-bonding liquid at low-to- moderate pressure → an activity-coefficient model (NRTL/UNIQUAC); non-polar / hydrocarbon / high-pressure gas → a cubic EoS (PR/SRK); water + gas at high pressure → an associating model (CPA); aqueous salts → an electrolyte model. MaximaLabs encodes that judgment: it looks at your component list and, if the active method is a poor fit, surfaces a one-click **recommendation**. Put a polar ethanol-water mixture on Peng-Robinson and this banner appears over the canvas — the decision tree, made actionable, before a wrong answer is ever computed:



The property-method recommender: on a polar ethanol-water mixture set to Peng-Robinson, MaximaLabs flags the mismatch and offers a one-click switch to NRTL — the decision tree, made actionable.

2.6 Seeing it: the phase diagram

The honest test of a property method is whether it reproduces measured phase equilibrium. MaximaLabs plots the binary **T-xy** / P-xy diagram straight from the active thermo package, so you can look at the VLE before trusting a column. For ethanol-water under NRTL, the vapor and liquid curves pinch together and cross the diagonal at the azeotrope — exactly the feature a cubic EoS misses:

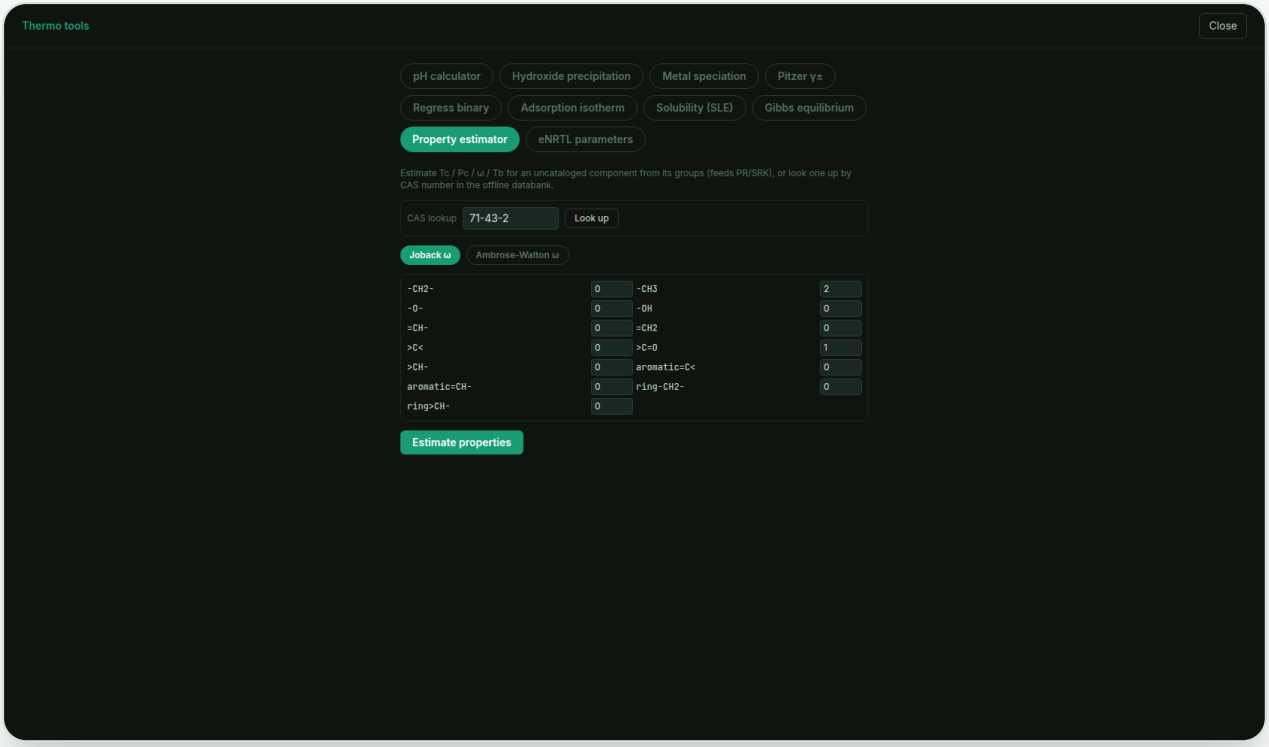


The ethanol-water T-xy phase diagram under NRTL. The bubble/dew curves meet the diagonal at the azeotrope — the ceiling ordinary distillation can't cross. MaximaLabs validates this system against published DECHEMA VLE data.

This is also where trust comes from: MaximaLabs pins the ethanol-water NRTL prediction against published data on its validation dashboard (bubble-point AAD < 0.2 K, azeotrope at x ≈ 0.894 / 351.3 K).

2.7 When a component isn't in the databank

Real work runs into molecules the databank has never heard of. Rather than stop, you estimate the missing pure-component properties from structure — group contribution (Joback) for critical constants and normal boiling point, Ambrose-Walton for vapor pressure. MaximaLabs does this in the Thermo tools panel — **Analysis & reports ▸ Thermo tools ▸ Property estimator**. Enter the molecular groups (or look the component up by CAS number) and it returns the critical constants, normal boiling point, and acentric factor; the estimated component then drops straight into the same flowsheet:



Thermo tools ▸ Property estimator: Joback / Ambrose-Walton group-contribution estimates (or a CAS lookup) for an uncataloged component — critical constants, boiling point, acentric factor — usable in the very next solve.

2.8 Try it: watch the thermodynamics change the answer

REPRODUCE IT IN YOUR BROWSER

1. Open the ethanol-water distillation example and **Run** it (it defaults to NRTL). Note the distillate ethanol purity in the stream table.
2. Click the **NRTL** pill in the status bar and switch the property method to **Peng-Robinson**. Watch the recommender flag the mismatch and offer NRTL back.
3. **Run** again under Peng-Robinson and compare the distillate purity — the cubic misjudges the azeotrope, so the separation looks different from the NRTL answer.

4. Open **Plots** (right rail ▸ Analysis & reports) and generate the **Phase diagram** for ethanol/water under each package — the azeotrope is there under NRTL and distorted under the cubic.
5. Switch back to **NRTL** and re-run to restore the physically correct result.

The lesson of the chapter: the solver from Chapter 1 will faithfully converge *whatever* thermodynamics you give it — so the property method is yours to get right. Next up, the calculation every one of these models feeds: the flash.

2.9 Exercises

PRACTICE

Work each problem yourself first, then reveal the solution to check it. Where a problem says so, reproduce it live in MaximaLabs — the solver is the answer key.

1 WARM-UP

Pick the property method for each and justify in one line: (a) a high-pressure natural-gas mixture of methane, ethane, and propane; (b) an ethanol-water separation at atmospheric pressure.

Show solution

2 CORE

At an **azeotrope**, the vapor and liquid compositions are equal (

$$y_i = x_i$$

). What does the relative volatility

$$\alpha$$

equal there, and what does that imply for a simple column trying to push ethanol past its ~89 mol% azeotrope?

Show solution

3 CHALLENGE

Suppose you select **Peng-Robinson** for the ethanol-water column instead of NRTL. Predict what goes wrong, then check it: open ethanol-water distillation and switch the thermo package under the Thermodynamics menu.

Show solution

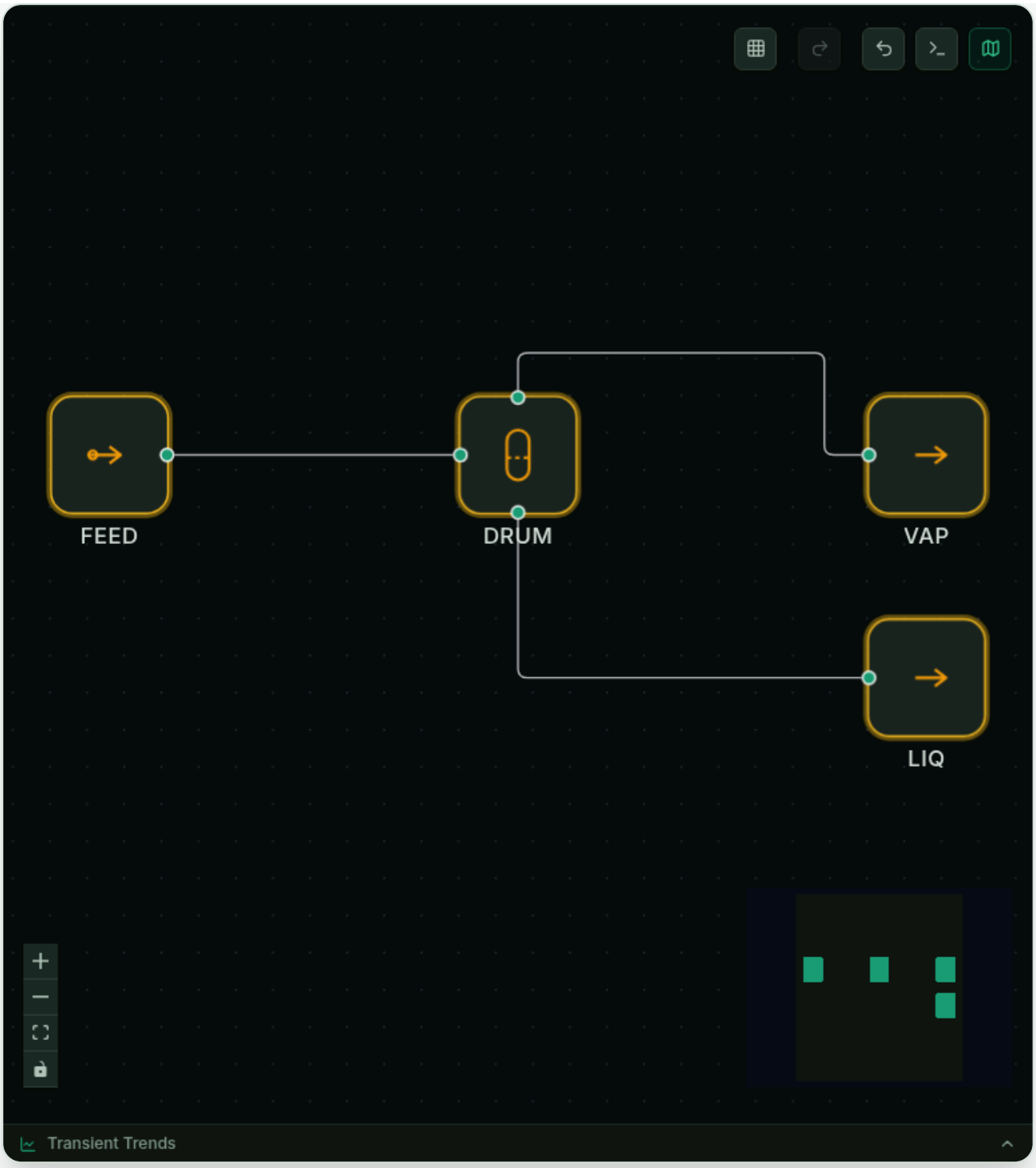
CHAPTER 3

Flash & Phase Equilibrium

Chapter 2 chose the thermodynamics. This chapter runs them. The **flash** — take a feed at some temperature and pressure and ask how it splits into vapor and liquid at equilibrium — is the single most-called calculation in a simulator. A distillation column flashes on every stage on every iteration; a recycle loop flashes thousands of times before it converges. Get the flash right and fast, and everything above it works.

3.1 One equilibrium stage

The simplest unit that *is* a flash is the flash drum: one feed in, a vapor product and a liquid product out, held at a set temperature and pressure. The reference example below splits an ethanol-water feed at 358 K into an ethanol-rich vapor and a water-rich liquid.



The 'Flash separation' example: FEED → flash DRUM → vapor (VAP) + liquid (LIQ). One equilibrium stage, the atom every column is built from.

3.2 The isothermal flash: Rachford-Rice

Given the feed composition

$$z_i$$

, temperature, and pressure, the flash finds the **vapor fraction**

$$\psi$$

(moles of vapor per mole of feed) and the phase compositions. Material balance on each component plus the equilibrium relation

$$y_i = K_i x_i$$

gives the phase mole fractions in terms of

$$\psi$$

:

$$x_i = \frac{z_i}{1 + \psi\,(K_i - 1)}, \qquad y_i = \frac{z_i\,K_i}{1 + \psi\,(K_i - 1)}$$

Requiring both phases' mole fractions to sum to one,

$$\sum_i (y_i - x_i) = 0$$

, collapses to a single equation in the single unknown

$$\psi$$

— the **Rachford-Rice** equation:

$$\sum_i \frac{z_i\,(K_i - 1)}{1 + \psi\,(K_i - 1)} = 0$$

z_i	Feed mole fraction of component i (known — this is the flash's input composition).
$x_i,\ y_i$	Liquid- and vapor-phase mole fraction of component i (unknowns being solved for).
K_i	Equilibrium ratio K_i = y_i/x_i — how strongly i favors the vapor (from §3.3).
ψ	Vapor fraction — moles of vapor per mole of feed (the one unknown Rachford-Rice solves for; every x_i, y_i follows from it).

It's monotonic in

$$\psi$$

, so a safeguarded root-find nails it in a handful of steps. A value

$$0 < \psi < 1$$

is a genuine two-phase mixture;

$$\psi \le 0$$

means the feed is a subcooled liquid,

$$\psi \ge 1$$

a superheated vapor.

WORKED EXAMPLE — THE FLASH-SEPARATION REFERENCE CASE

Feed

$$z = (\text{ethanol: } 0.4,\ \text{water: } 0.6)$$

at 358 K, 1 atm, under NRTL (§3.1's example). Solving Rachford-Rice with the thermo package's real K-values gives:

$$\psi = 0.759, \qquad K_{\text{ethanol}} = 3.69,\ \ K_{\text{water}} = 0.593$$

$$x = (\text{ethanol: } 0.132,\ \text{water: } 0.868), \qquad y = (\text{ethanol: } 0.485,\ \text{water: } 0.515)$$

Check it:

$$x_{\text{ethanol}} = z/(1 + \psi(K - 1)) = 0.4/(1 + 0.759 \times 2.69) \approx 0.132$$

— matches. 76% of the feed vaporizes (

$$\psi = 0.759$$

), and because

$$K_{\text{ethanol}} > 1 > K_{\text{water}}$$

, that vapor concentrates ethanol (48.5% vs. the feed's 40%) while the liquid concentrates water (86.8% vs. 60%) — this is the real number the **Try it** section below reproduces.

3.3 K-values, and why the flash iterates

The catch: the K-values

$$K_i = y_i / x_i$$

aren't constants — they depend on temperature, pressure, *and* the very compositions the flash is trying to find (through the property model from Chapter 2):

$$K_i = \frac{y_i}{x_i} = \frac{\gamma_i \, P_i^{\text{sat}}(T)}{\varphi_i^V \, P} \;\; (\text{activity model}) \qquad \text{or} \qquad K_i = \frac{\hat{\varphi}_i^{\,L}}{\hat{\varphi}_i^{\,V}} \;\; (\text{EoS})$$

γ_i	Liquid activity coefficient — the non-ideality correction Chapter 2's NRTL/UNIQUAC/Wilson models supply; a value of 1 is the ideal (Raoult's-law) limit.
$P_i^{\text{sat}}(T)$	Pure-component vapor pressure of i at the flash temperature.
φ_i^V	Vapor-phase fugacity coefficient (≈1 near atmospheric pressure; departs from 1 as pressure rises).
$\hat{\varphi}_i^{\,L}$, $\hat{\varphi}_i^{\,V}$	Liquid- and vapor-phase fugacity coefficients of i in the mixture, from an equation of state (PR/SRK) — the EoS route, used instead of an activity coefficient when there's no natural liquid/vapor asymmetry (e.g. near-critical or all-hydrocarbon systems).

So the flash is a loop: start from a cheap K-value estimate (the Wilson correlation, which needs only critical constants and the acentric factor),

$$\ln K_i \approx \ln \frac{P_{c,i}}{P} + 5.373\,(1+\omega_i)\left(1 - \frac{T_{c,i}}{T}\right)$$

$P_{c,i}$, $T_{c,i}$	Critical pressure and temperature of component i — fixed physical constants, not fitted per-mixture.
ω_i	Acentric factor of i (how much its vapor-pressure curve deviates from a simple fluid's — near 0 for small symmetric molecules like methane, larger for bigger/polar ones).

solve Rachford-Rice for

$$\psi$$

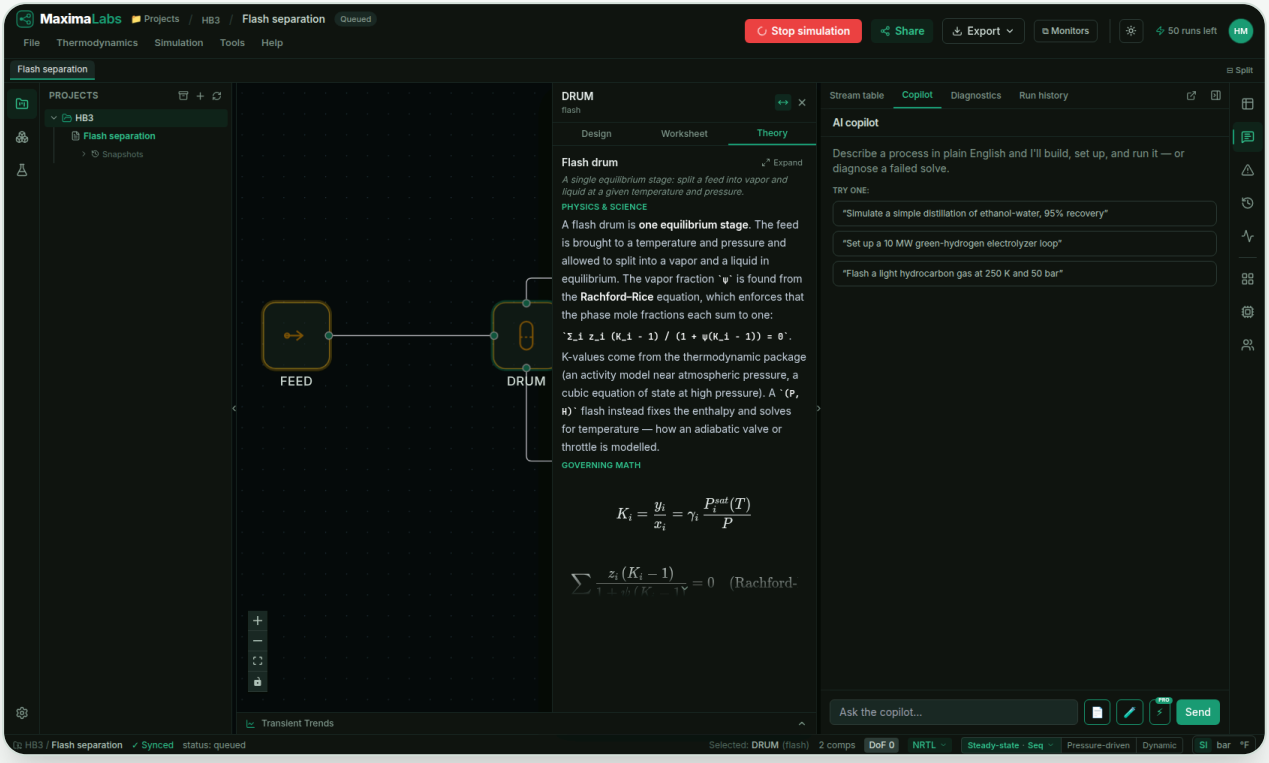
, recompute

$$x_i, y_i$$

, ask the thermo package for updated K-values, and repeat (successive substitution), finishing with a Newton polish near the solution. This is exactly the loop MaximaLabs runs — and it will show you the equations it's solving.

3.4 See the equations being solved

Click the flash drum, open the **Theory** tab, and MaximaLabs renders the drum's governing equations — the K-value relation and Rachford-Rice — in the same panel as its solved values. Aspen solves this; here you can read it:



The flash drum's 'show the math' (ParamEditor ▸ Theory): the plain-English mechanism plus the KaTeX K-value and Rachford-Rice equations. The same panel notes that a (P,H) flash fixes enthalpy and solves for temperature — how a valve or throttle is modelled.

3.5 Bubble and dew points

Two limits of the flash are worth naming because whole design specs hang on them. The **bubble point** is where the first bubble of vapor forms —

$$\psi \to 0$$

, so Rachford-Rice reduces to

$$\sum_i K_i \, z_i = 1 \quad \text{(bubble point)}$$

and the **dew point** is where the first drop of liquid condenses —

$$\psi \to 1$$

:

$$\sum_i \frac{z_i}{K_i} = 1 \quad \text{(dew point)}$$

These are what the phase diagram of Chapter 2 traces out: the bubble curve and the dew curve. Solving for the T that satisfies each (at fixed P) is a one-dimensional root-find on the same K-values.

WORKED EXAMPLE — BUBBLE POINT OF THE SAME FEED

Take

$$z = (\text{ethanol: } 0.4,\ \text{water: } 0.6)$$

at 1 atm again and ask for its bubble temperature instead of fixing 358 K. MaximaLabs' NRTL package solves

$$\sum_i K_i(T) \, z_i = 1$$

for T and returns

$$T_{\text{bubble}} \approx 353.7 \text{ K}$$

— below 358 K, consistent with §3.1's example already being partially vaporized (

$$\psi = 0.759 > 0$$

) at that hotter condition.

3.6 The adiabatic (P,H) flash

Not every flash fixes temperature. When a stream is throttled through a valve, or two streams mix adiabatically, what's conserved is *enthalpy*, not T. The **(P,H) flash** fixes pressure and total enthalpy and solves for temperature and the split together — an outer energy balance wrapped around the same Rachford-Rice inner loop:

$$H_{\text{feed}} = \psi\, H^V(T) + (1-\psi)\, H^L(T)$$

H_{feed}	Total feed enthalpy — known and fixed (what's conserved across the valve/mixer).
$H^V(T),\ H^L(T)$	Vapor- and liquid-phase molar enthalpy at the (unknown) outlet temperature T, from the thermo package.
$\psi,\ T$	The two unknowns solved simultaneously: vapor fraction and outlet temperature (T also sets every K_i inside the inner Rachford-Rice solve, which is why this wraps the isothermal flash rather than replacing it).

That temperature drop across a valve — Joule-Thomson cooling — falls straight out of this. In MaximaLabs a flash drum with a duty instead of a temperature spec (duty = 0 for adiabatic) runs exactly this calculation.

3.7 Try it

REPRODUCE IT IN YOUR BROWSER

1. Open the flash separation example and **Run** it.
2. Read the **Stream table**: the vapor product is ethanol-enriched and the liquid water-enriched — one stage of separation, driven purely by the difference in volatility the K-values capture.
3. Click the **DRUM** node and open the **Theory** tab to see the Rachford-Rice and K-value equations it just solved.
4. Open the flash drum's params and change the **temperature** (try 350 K vs. 365 K) and re-run — watch the vapor fraction and the split move as you cross further into the two-phase region.
5. For the adiabatic case, switch the drum's spec from temperature to **duty = 0** and re-run — now it solves for the temperature that balances enthalpy.

With a reliable flash in hand, the next chapters just stack it up: a reactor flashes its outlet, and a distillation column is a whole column of these equilibrium stages wired together.

3.8 Exercises

PRACTICE

Work each problem yourself first, then reveal the solution to check it. Where a problem says so, reproduce it live in MaximaLabs — the solver is the answer key.

1 WARM-UP

A binary feed

$$z = (0.5,\ 0.5)$$

is held at conditions where the K-values are

$$K_1 = 2.5$$

and

$$K_2 = 0.4$$

. Without solving Rachford-Rice, decide whether the feed is **two-phase**.

Show solution

2 CORE

A feed

$$z = (0.3,\ 0.7)$$

is exactly at its **bubble point** at some temperature where the light key has

$$K_1 = 2.0$$

. What K-value must the heavy component have?

Show solution

3 CHALLENGE

A warm liquid is throttled across a valve to a lower pressure and partially vaporizes. Using the

$$(P, H)$$

flash, explain why its temperature *falls*, then reproduce the effect in the flash separation example.

Show solution

CHAPTER 4

Reactors

Everything so far moved molecules around and split them by phase. A reactor is the first node that *changes what the molecules are*. How you model it depends on what you know: sometimes you know the reaction and how far it goes, sometimes only that it reaches equilibrium, sometimes not even which reactions occur. MaximaLabs gives you four reactor models for exactly that spread of knowledge.

4.1 Four models, four things you might know

In rough order of how much you have to supply:

- **Stoichiometric / conversion** — you know the reaction and set how far it goes (a fixed conversion of a key component).
- **Equilibrium** — you know the reaction but let thermodynamics set how far it goes, from an equilibrium constant.
- **Gibbs** — you don't specify reactions at all; you give the possible species and let free-energy minimization find the equilibrium mix.
- **Kinetic (CSTR / PFR)** — you have a rate law, so reactor *size* and residence time set the conversion.

4.2 The stoichiometric reactor

The most direct model: give the stoichiometry

$$\nu_i$$

(negative for reactants, positive for products), pick a key component, and set its fractional conversion

$$X_{key}$$

. Everything follows from the **extent of reaction**

$$\xi$$

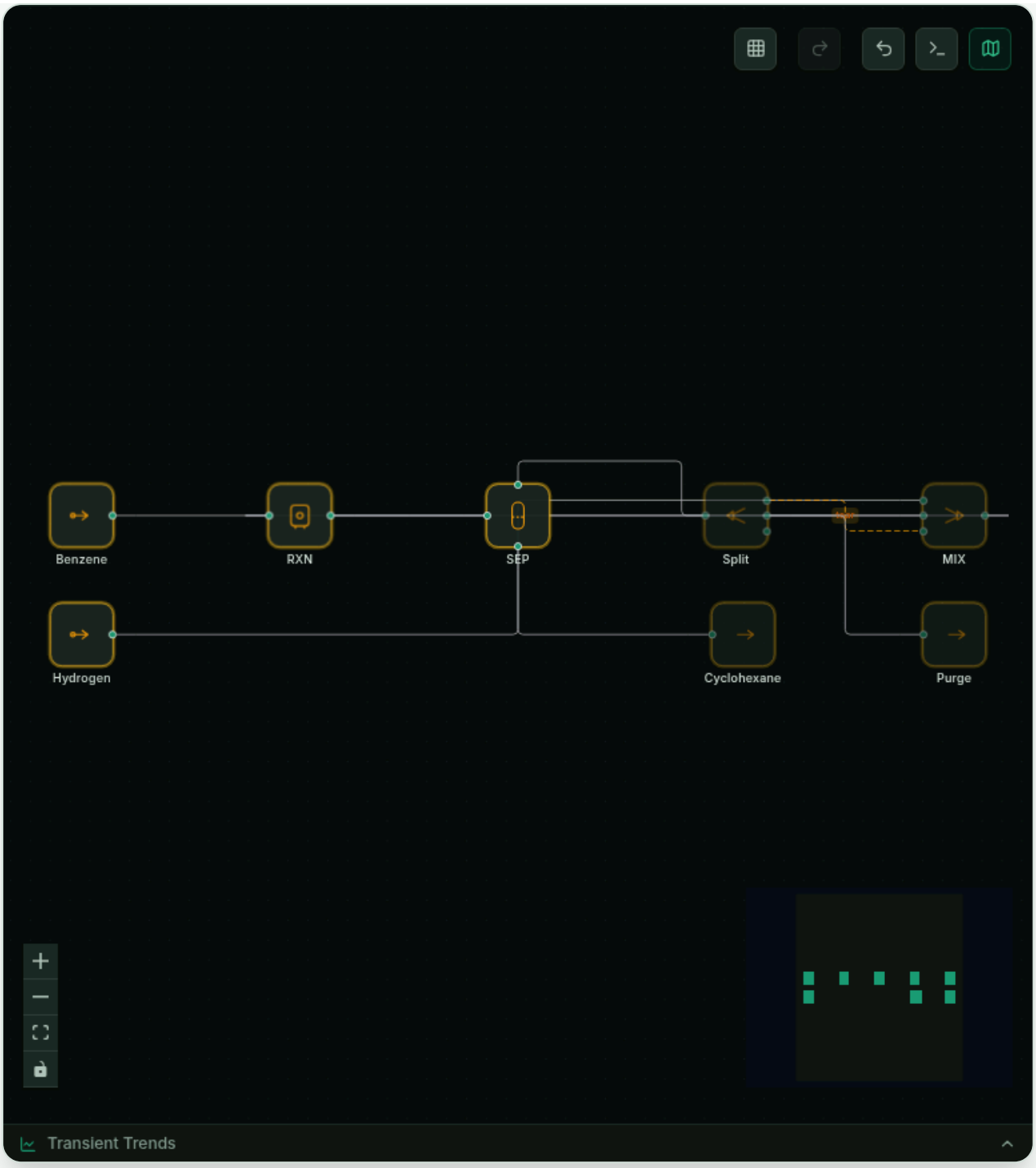
:

$$\xi = X_{key}\,\frac{F\,z_{key}}{|\nu_{key}|}, \qquad \dot{n}_{i,\text{out}} = \dot{n}_{i,\text{in}} + \nu_i\,\xi$$

The example below is benzene hydrogenation —

$$\mathrm{C_6H_6} + 3\,\mathrm{H_2} \rightarrow \mathrm{C_6H_{12}}$$

— run at 97% conversion of benzene, with the unreacted hydrogen flashed off and recycled and a small purge to keep inerts from building up. A reactor almost never stands alone; it lives in a loop, exactly the recycle machinery from Chapter 1.



The 'Benzene hydrogenation' example: two feeds → mixer → conversion reactor → flash → recycle + purge. The reactor is one node; the process around it closes the hydrogen balance.

4.3 The heat of reaction

Reactions release or absorb energy, and the reactor's energy balance carries it. For extent

$$\xi$$

and an added duty

$$Q$$

:

$$F_{\text{out}}\, h_{\text{out}} = F_{\text{in}}\, h_{\text{in}} - \Delta H_{\text{rxn}}\, \xi + Q$$

Run it adiabatically (

$$Q = 0$$

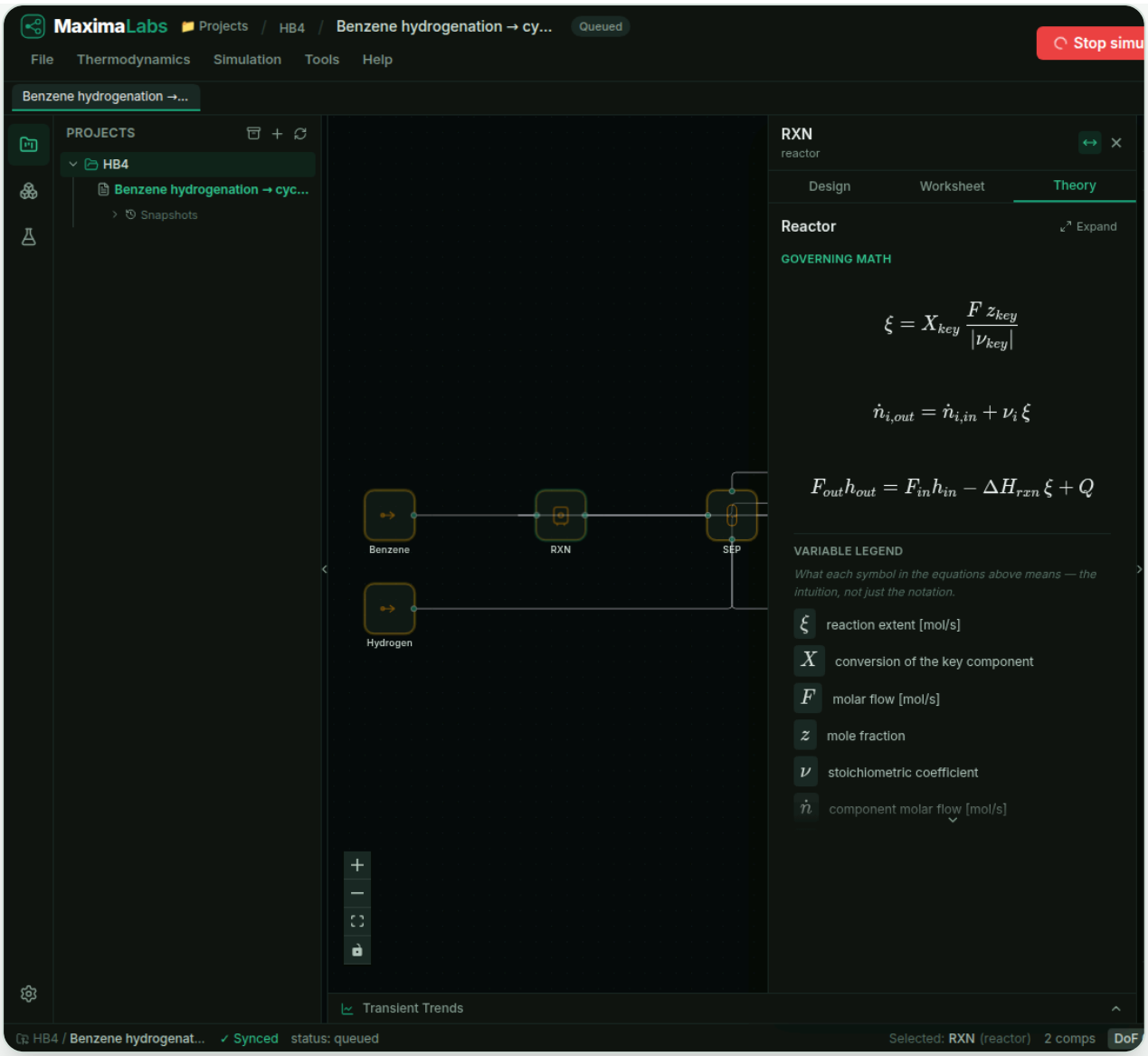
) and that

$$\Delta H_{\text{rxn}}$$

shows up as a temperature change — the adiabatic temperature rise of an exothermic reaction, or the chill of an endothermic one. Hydrogenation is strongly exothermic, so the reactor outlet runs hot; the downstream flash and its temperature spec are there to manage it.

4.4 See the math

As with every unit, click the reactor and open **Theory** to see the exact equations — the extent of reaction, the per-component mole balance, and the heat-of-reaction energy balance — rendered alongside the reactor's solved conversion and duty:



The reactor's 'show the math' (ParamEditor ▸ Theory): extent of reaction ξ, the mole balance ṅ_out = ṅ_in + νξ, and the energy balance with ΔH_rxn — the equations MaximaLabs just solved for this reactor.

4.5 The equilibrium reactor

When a reaction runs to thermodynamic equilibrium rather than a fixed conversion, you supply the reaction and an equilibrium constant, and MaximaLabs solves for the extent that satisfies the law of mass action:

$$\prod_i a_i^{\nu_i} = K_{eq}(T)$$

The temperature dependence of

$$K_{eq}$$

comes from the van't Hoff relation, so an exothermic reaction's equilibrium backs off as it heats up (Le Chatelier, quantified):

$$\ln\frac{K_{eq}(T)}{K_{ref}} = -\frac{\Delta H_{\text{rxn}}}{R}\left(\frac{1}{T} - \frac{1}{T_{ref}}\right)$$

This is exactly what the ammonia synthesis loop from Chapter 1 uses — only ~10% converts per pass because equilibrium (not conversion) caps it, which is exactly why it needs the big recycle.

4.6 The Gibbs reactor

Sometimes you don't know the reactions — combustion and reforming produce dozens of species through networks no one writes out by hand. The Gibbs reactor sidesteps that entirely: give it the candidate species and it finds the composition that **minimizes total Gibbs free energy** subject to conserving atoms:

$$\min_{n_i} \frac{G}{RT} = \sum_i n_i \left[\frac{\Delta G^\circ_{f,i}(T)}{RT} + \ln\frac{n_i}{n_{tot}} + \ln\frac{P}{P^\circ}\right]$$

$$\text{subject to} \quad \sum_i n_i\, a_{ij} = b_j \quad (\text{atoms of element } j), \qquad n_i \geq 0$$

n_i	Moles of candidate species i at the outlet — the unknowns the minimization solves for.
G/RT	Total Gibbs free energy of the mixture (dimensionless, scaled by RT) — the quantity being minimized.
$\Delta G^\circ_{f,i}(T)$	Standard Gibbs energy of formation of species i at temperature T — a tabulated thermodynamic property, not a fitted parameter.
a_{ij}	Number of atoms of element j in one molecule of species i — fixed by each species' chemical formula.
b_j	Total moles of element j available (from the feed) — conserved regardless of which reactions actually occur.

No reaction equations required — just thermodynamics and a mass balance. It's the natural model for a reformer, a combustor, or a syngas equilibrium.

WORKED EXAMPLE — SO_2 OXIDATION IN A SULFURIC-ACID CONTACT PROCESS

Roaster gas

$$z = (\mathrm{SO_2}\colon 0.08,\ \mathrm{O_2}\colon 0.11,\ \mathrm{N_2}\colon 0.81)$$

at 100 mol/s, 700 K, 2 atm, into a Gibbs reactor given the candidate species

$$\{\mathrm{SO_2}, \mathrm{SO_3}, \mathrm{O_2}, \mathrm{N_2}\}$$

— no reaction written down, just the species list. Minimizing free energy at 700 K (where SO_2 oxidation is strongly favored) converts nearly all of it:

$$\dot{n}_{\mathrm{SO_2,in}} = 8.0\ \mathrm{mol/s}, \qquad \dot{n}_{\mathrm{SO_2,out}} = 96.03 \times 0.00071 \approx 0.068\ \mathrm{mol/s}$$

A conversion of

$$(8.0 - 0.068)/8.0 \approx 99.2\%$$

— the equilibrium composition MaximaLabs' Gibbs solver finds directly from thermodynamics, with no SO_2+½O_2→SO_3 reaction ever specified.

4.7 The kinetic reactor (CSTR / PFR)

When conversion is set by *rate* and *residence time* rather than equilibrium, use the kinetic reactor. You give an Arrhenius rate law and a reactor volume, and the model solves the CSTR or PFR design equation for the outlet:

$$k(T) = k_0\ \exp\!\left(-\frac{E_a}{RT}\right), \qquad r = k(T)\prod_i c_i^{m_i}$$

$k(T)$	Rate constant at the reactor temperature (units depend on the overall order) — grows with temperature per Arrhenius.
k_0	Pre-exponential (frequency) factor — the rate constant's theoretical ceiling as T→∞.
E_a	Activation energy (J/mol) — the energy barrier the reaction must clear; a bigger E_a makes the rate more temperature-sensitive.
$R,\ T$	Gas constant (8.314 J/mol/K) and absolute reactor temperature (K).

r	Reaction rate (mol/volume/time) — what the CSTR/PFR design equation integrates against the residence time to get conversion.
$c_i,\ m_i$	Molar concentration of component i and its reaction order — the exponents you supply per the rate law's kinetics.

A CSTR is one well-mixed volume (outlet = tank composition); a PFR integrates the rate along the reactor length. Either way, the size you draw becomes the conversion you get. The refinery hydrocracker example uses a lumped first-order kinetic network in exactly this form.

WORKED EXAMPLE — A KINETIC CSTR

Feed

$$z = (\text{ethanol: } 0.5,\ \text{water: } 0.5)$$

at 5.0 mol/s, 340 K, into a 0.02 m³ CSTR with

$$k_0 = 1.0 \times 10^{6}\ \text{s}^{-1}$$

,

$$E_a = 40{,}000\ \text{J/mol}$$

, first-order in ethanol. The rate constant at 340 K:

$$k(340) = 1.0 \times 10^{6} \times \exp\left(-\frac{40{,}000}{8.314 \times 340}\right) \approx 0.715\ \text{s}^{-1}$$

Solving the CSTR's well-mixed material balance with this rate constant, MaximaLabs reports

$$98.6\%$$

conversion of ethanol (extent

$$\xi \approx 2.47\ \text{mol/s}$$

) at this volume and residence time — high conversion because the rate constant is fast (~0.7 s^{-1}) relative to the tank's holding time, not because a conversion was specified anywhere.

4.8 Try it

REPRODUCE IT IN YOUR BROWSER

1. Open the benzene hydrogenation example and **Run** it.
2. Click the **RXN** reactor and read its params: the stoichiometry (−1 benzene, −3 H_2, +1 cyclohexane), the key component, and the 0.97 conversion. Open **Theory** to see the extent and energy-balance equations.
3. In the **Stream table**, confirm the reactor outlet is nearly all cyclohexane and the leftover hydrogen carries forward to the flash.
4. Drop the conversion to **0.80** and re-run — more unreacted benzene survives, and the recycle/purge rebalances. Watch how the loop from Chapter 1 absorbs the change.
5. For the equilibrium contrast, open the ammonia loop — same idea, but thermodynamics (not a fixed conversion) sets how far it goes.

Pick the reactor model that matches your knowledge — a conversion you measured, an equilibrium you trust, a species list, or a rate law — and the solver does the rest. Next: separations, where the flash of Chapter 3 becomes a whole column.

4.9 Exercises

PRACTICE

Work each problem yourself first, then reveal the solution to check it. Where a problem says so, reproduce it live in MaximaLabs — the solver is the answer key.

1 WARM-UP

Ammonia synthesis runs

$$N_2 + 3\,H_2 \rightarrow 2\,NH_3$$

. A feed of 100 mol/s N_2 (with stoichiometric H_2) reacts at 30% N_2 conversion. Find the extent of reaction and the NH_3 produced.

Show solution

2 CORE

A reaction is **exothermic** (

$$\Delta H_{\text{rxn}} < 0$$

). Using van't Hoff,

$$\frac{d \ln K}{dT} = \frac{\Delta H_{\text{rxn}}}{RT^2}$$

, which way does the equilibrium constant

$$K$$

move as temperature rises, and what dilemma does that create for ammonia synthesis?

Show solution

3 CHALLENGE

For a normal positive-order reaction, does a **CSTR** or a **PFR** need more volume to reach the same conversion? Explain from the rate, then check it by dropping both reactor types with the same kinetics on the canvas and comparing the required volumes.

Show solution

CHAPTER 5

Separation Processes

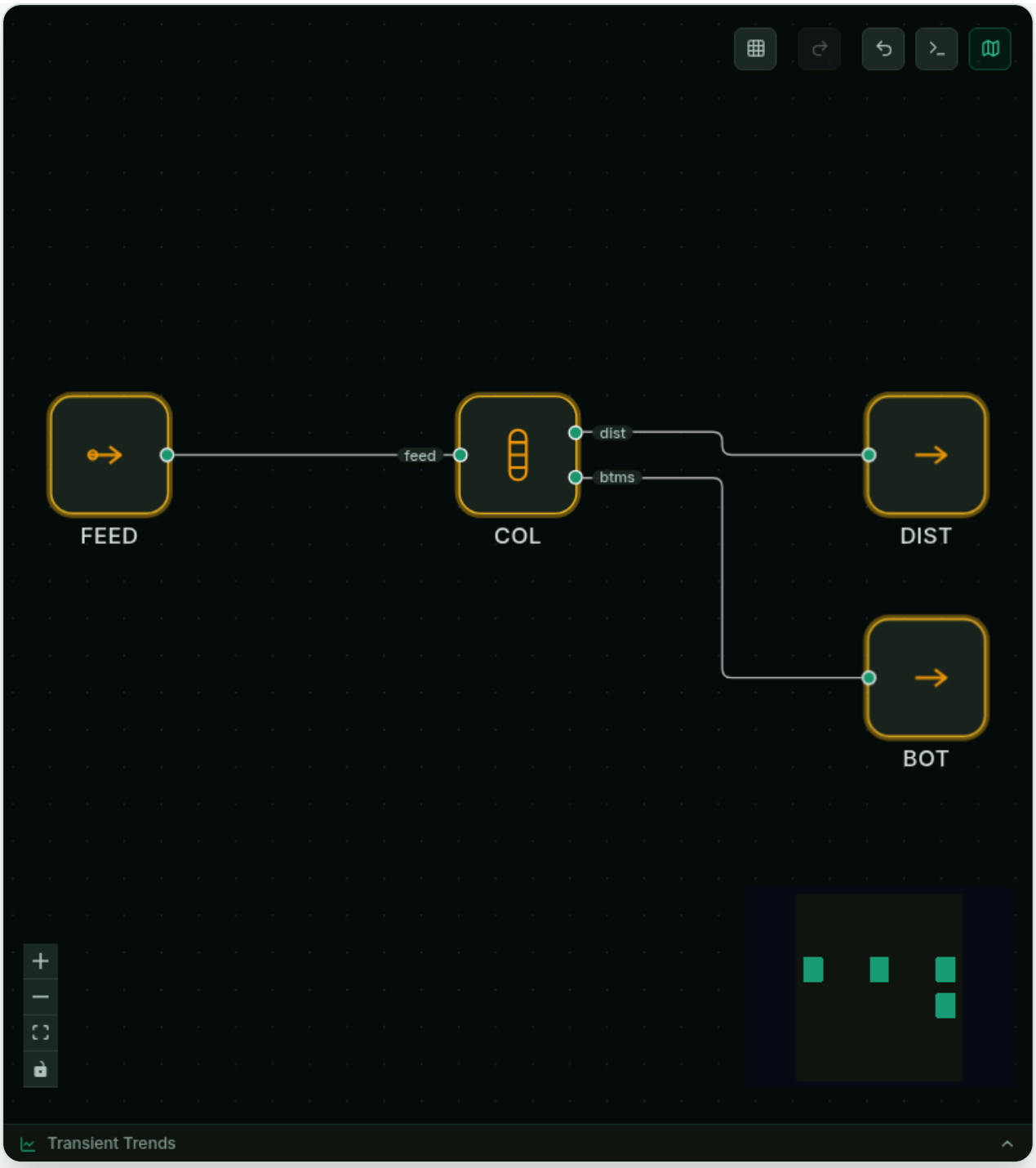
One flash (Chapter 3) buys you one stage of separation — often not enough. Stack many equilibrium stages, add reflux at the top and a reboiler at the bottom, and you get distillation: the workhorse that does most of the separation in the chemical industry. This chapter is how a column is modelled, how to read what it's doing, and how to size one fast.

5.1 A column is a stack of equilibrium stages

Picture the flash of Chapter 3 repeated on every tray: liquid falls down the column, vapor rises up it, and on each stage the two come to equilibrium — vapor enriching in the light component as it climbs, liquid enriching in the heavy component as it descends. A **condenser** at the top returns part of the overhead as reflux (ratio

$$R = L/D$$

), and a **reboiler** at the bottom boils part of the bottoms back up. The example below is the ethanol-water column from Chapter 1, now the object of study.



The ethanol-water distillation column: FEED → COL → distillate (DIST) + bottoms (BOT). Inside that one node is a whole stack of equilibrium stages.

5.2 The MESH equations

Rigorously, every stage

$$j$$

is described by four kinds of equation — the **MESH** equations — written for each component

$$i$$

:

$$\mathbf{M}\quad l_{i,j-1} + v_{i,j+1} + f_{i,j} - l_{i,j} - v_{i,j} = 0 \quad \text{(material balance)}$$

$$\mathbf{E}\quad y_{i,j} = K_{i,j}\, x_{i,j} \quad \text{(equilibrium)}$$

$$\mathbf{S}\quad \sum_{i} y_{i,j} - 1 = 0 \quad \text{(summation)}$$

$$\mathbf{H}\quad L_{j-1}h^{L}_{j-1} + V_{j+1}h^{V}_{j+1} + F_j h^{F} - L_j h^{L}_{j} - V_j h^{V}_{j} - Q_j = 0 \quad \text{(enthalpy)}$$

$l_{i,j},\ v_{i,j}$	Liquid- and vapor-phase molar flow of component i leaving stage j (the unknowns the M equations balance).
$f_{i,j}$	Feed flow of component i entering stage j (zero on every stage except the feed tray).
$y_{i,j},\ x_{i,j}$	Vapor and liquid mole fraction of component i on stage j.
$K_{i,j}$	Equilibrium ratio of component i on stage j (the same K from Chapter 3's flash, evaluated at that stage's T, P, composition).
$L_j,\ V_j$	Total liquid and vapor molar flow leaving stage j.
$h^{L}_{j},\ h^{V}_{j}$	Liquid- and vapor-phase molar enthalpy on stage j.
Q_j	Heat added to (or removed from) stage j — nonzero only at the condenser and reboiler for an ordinary column.

That's

$$M \times N$$

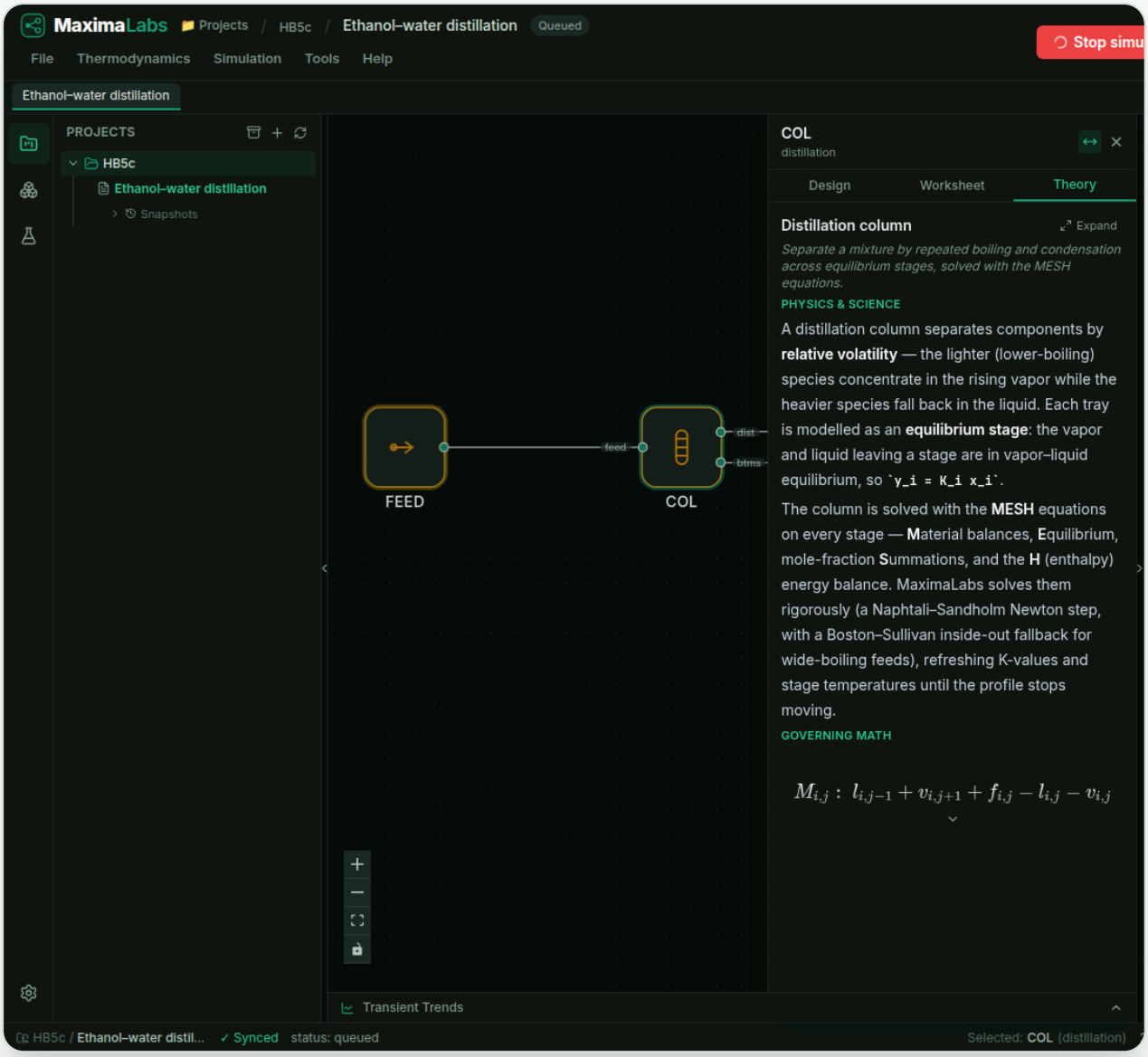
coupled nonlinear equations for an

$$N$$

-stage,

M

-component column, solved simultaneously by a Newton method — the equation-oriented idea from Chapter 1, applied inside a single unit. MaximaLabs shows you the exact set on the column's Theory tab:



The column's 'show the math' (ParamEditor ▸ Theory): the MESH equations MaximaLabs solves on every stage, rendered in full — the same equations Aspen's RadFrac solves, here in plain view.

5.3 McCabe-Thiele: the picture behind the column

Before rigorous solvers existed, engineers sized binary columns graphically with the **McCabe-Thiele** construction — and it's still the clearest picture of what a column does. On the equilibrium

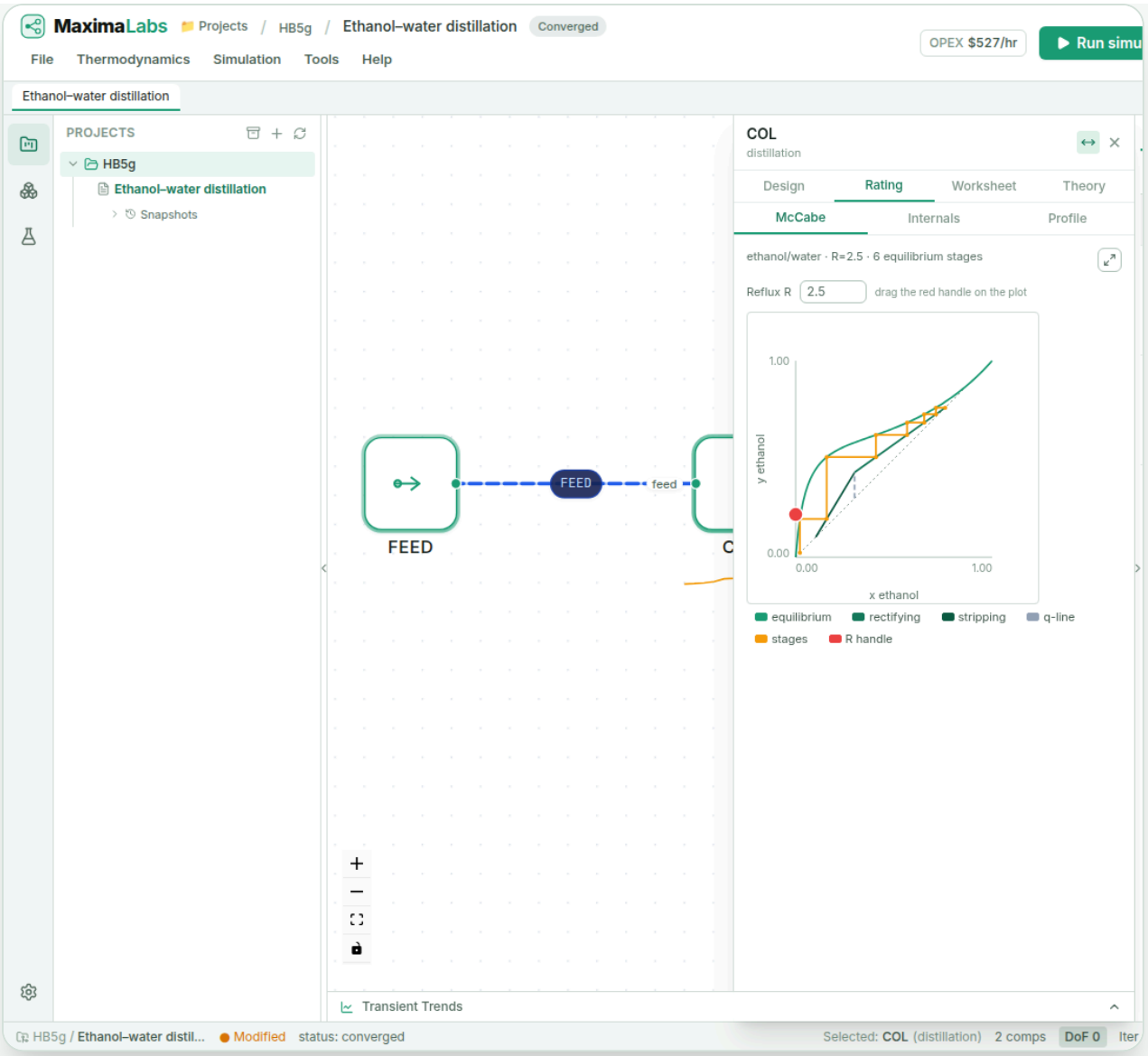
x-y

diagram from Chapter 2 you draw two *operating lines* (the mass balances above and below the feed),

$$y_{n+1} = \frac{R}{R+1}\,x_n + \frac{x_D}{R+1} \quad \text{(rectifying operating line)}$$

R	Reflux ratio L/D — moles refluxed per mole of distillate.
$x_n,\ y_{n+1}$	Liquid composition leaving stage n and vapor composition entering it from below — successive points on the operating line.
x_D	Distillate composition (the line's fixed point: at x_n=x_D, y_{n+1}=x_D too).

and step off stages as a staircase between the operating lines and the equilibrium curve — each tread is one theoretical stage. MaximaLabs draws it for a solved column, actual stages and all:



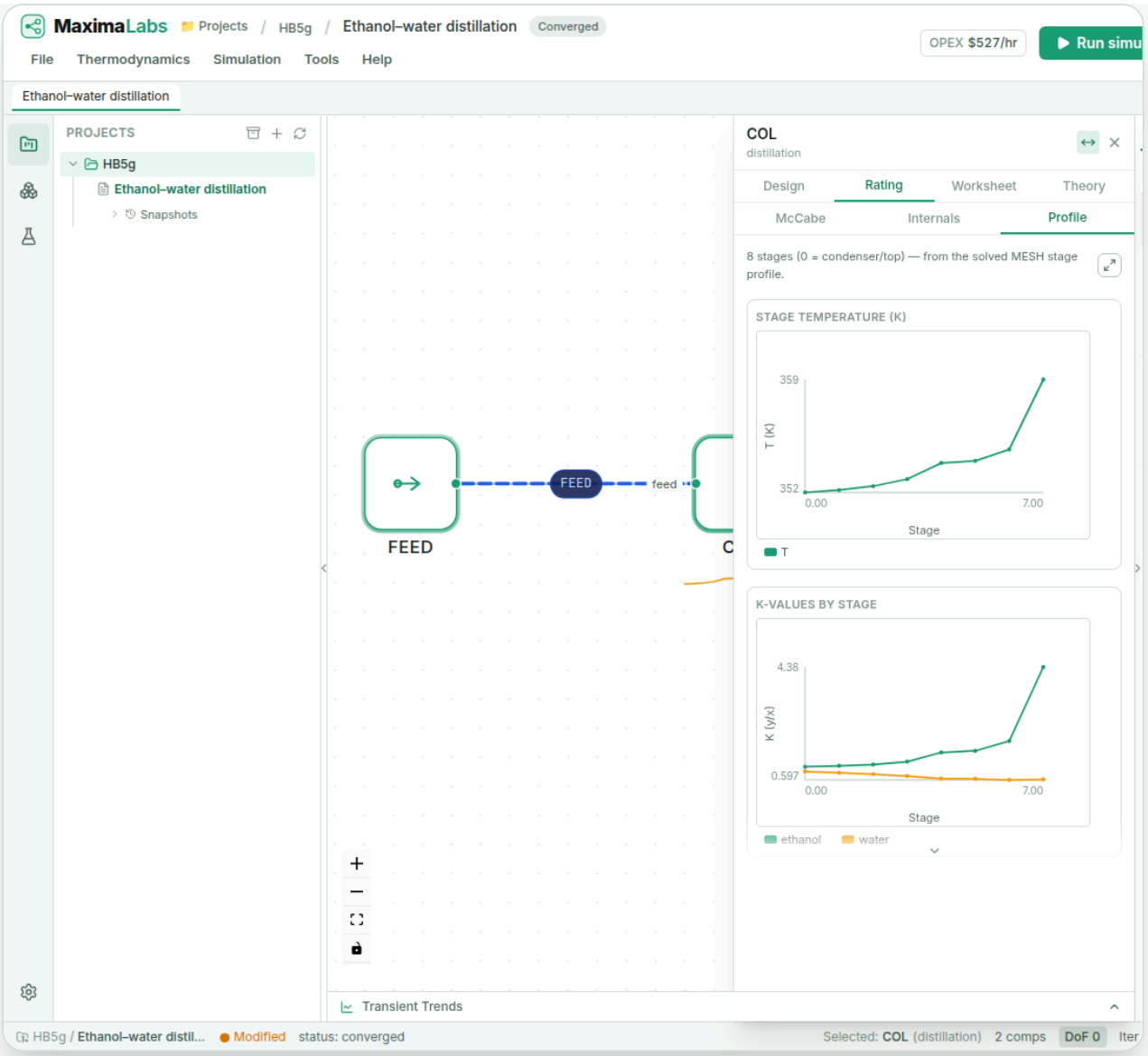
McCabe-Thiele for the ethanol-water column: the equilibrium curve, the operating lines set by the reflux ratio, and the step-off staircase — one step per theoretical stage. The steps bunch up as they approach the azeotrope.

5.4 Reading the column: stage profiles

A rigorous solve gives more than top and bottom products — it gives the full internal state, stage by stage: temperature, the liquid and vapor traffic (

$$L_j, V_j$$

), and the composition on every tray. These **profiles** are how you diagnose a column — spot a pinch (stages doing no separation), find the real feed-stage optimum, or see a temperature break:



The column's stage profiles (ParamEditor ▸ Profile): temperature and internal liquid/vapor flows down the column. Where the temperature curve flattens, stages are doing little work — a pinch.

5.5 Shortcut design (Fenske-Underwood-Gilliland)

A full MESH solve needs a column to already exist. To *design* one — how many stages, what reflux — engineers start with the **FUG shortcut** (Aspen's DSTWU). Fenske gives the minimum stages at total reflux from the relative volatility

$$\alpha$$

and the desired split:

$$N_{\min} = \frac{\ln\left[\left(\frac{x_{LK}}{x_{HK}}\right)_D \left(\frac{x_{HK}}{x_{LK}}\right)_B\right]}{\ln \alpha_{LK,HK}}$$

$N_{\min}$	Minimum theoretical stages, at total reflux (R=infinity) — the fewest stages any design at this split could use.
$\alpha_{LK,HK}$	Relative volatility of the light key over the heavy key.
$(x_{LK}/x_{HK})_D,\ (x_{HK}/x_{LK})_B$	Light-to-heavy-key ratio in the distillate, and heavy-to-light-key ratio in the bottoms — how sharp the desired split is.

WORKED EXAMPLE — FENSKE MINIMUM STAGES

$$\alpha = 2.4$$

, distillate 95 mol% light key, bottoms 5 mol% light key (the same case as §5.8's warm-up exercise):

$$N_{\min} = \frac{\ln[(0.95/0.05)(0.95/0.05)]}{\ln 2.4} = \frac{\ln 361}{\ln 2.4} \approx 6.7$$

About 7 theoretical stages at total reflux — the floor a real, finite-reflux design sits above.

Underwood gives the minimum reflux ratio — the other end of the design-space wall, where an infinite number of stages would be needed. For a multicomponent feed you first solve for the root(s)

$$\theta$$

of

$$\sum_i \frac{\alpha_i\, x_{F,i}}{\alpha_i - \theta} = 1 - q \qquad (\alpha_{HK} < \theta < \alpha_{LK})$$

$\alpha_i,\ x_{F,i}$	Relative volatility and feed mole fraction of component i.
θ	The Underwood root(s) — a value strictly between the heavy- and light-key volatilities, solved for numerically.
q	Feed liquid fraction (thermal condition): q=1 saturated liquid, q=0 saturated vapor, in between a flashed feed.

where

$$q$$

is the feed's liquid fraction (thermal condition), then use

$$\theta$$

in a second summation over the distillate to get

$$R_{\min}$$

:

$$R_{\min} + 1 = \sum_i \frac{\alpha_i \, x_{D,i}}{\alpha_i - \theta}$$

$R_{\min}$	Minimum reflux ratio — the other design-space wall (infinite stages needed below this).
$x_{D,i}$	Distillate mole fraction of component i.

Gilliland then closes the loop: given

$$N_{\min}$$

(Fenske) and

$$R_{\min}$$

(Underwood), pick an actual operating reflux

$$R \approx 1.2\text{–}1.5 \, R_{\min}$$

and read off the actual stage count

$$N$$

from an empirical correlation between the two dimensionless ratios:

$$\frac{N - N_{\min}}{N + 1} = 1 - \exp\left[\left(\frac{1 + 54.4X}{11 + 117.2X}\right)\left(\frac{X - 1}{\sqrt{X}}\right)\right], \qquad X = \frac{R - R_{\min}}{R + 1}$$

N	Actual stage count at the chosen operating reflux R — what the correlation solves for.
X	The dimensionless reflux ratio (R-R_min)/(R+1) — Gilliland's correlating variable.

WORKED EXAMPLE — A REAL FUG SHORTCUT RUN

A propane/n-butane depropanizer (50/50 feed, 8 atm, saturated liquid, 99% light-key recovery/99% heavy-key rejection), solved via MaximaLabs' actual shortcut solver under Peng-Robinson:

$$\alpha_{avg} = 2.91, \quad N_{\min} = 8.60, \quad R_{\min} = 1.01$$

$$R = 1.31\,R_{\min} = 1.31, \quad N \approx 19.1 \text{ stages}, \quad \text{feed stage} \approx 9.6$$

Real solver output, not hand-picked — this is exactly what the Shortcut column tool below returns for this feed.

Fenske → Underwood → Gilliland is the classic one-shot sequence: three cheap algebraic equations instead of a full MESH solve, good enough to size a column before committing to the rigorous model. MaximaLabs exposes exactly this as the **Shortcut column** tool (Analysis & reports ▸ Engineering analysis) — enter the light/heavy key recoveries and it returns

$$N_{\min}$$

,

$$R_{\min}$$

, and the Gilliland-estimated actual

$$N$$

in one call, the same three equations above.

5.6 The wall: azeotropes

Distillation separates by relative volatility, so where volatility vanishes —

$$\alpha \to 1$$

, the azeotrope — an ordinary column stalls. On the McCabe-Thiele picture the staircase can't step past the point where the equilibrium curve meets the diagonal; ethanol-water pinches at about 89 mol% ethanol no matter how many trays you add. Breaking that wall (pressure-swing, entrainer, extractive or azeotropic distillation) is its own craft — and exactly the sort of thing you'd ask the copilot to set up.

5.7 Try it

REPRODUCE IT IN YOUR BROWSER

1. Open the ethanol-water distillation example and **Run** it.
2. Click the **COL** node and open the **McCabe** tab to see the staircase, then **Profile** for the stage-by-stage temperature and flows, and **Theory** for the MESH equations.
3. Change the **reflux ratio** and re-run — watch the operating lines tilt on McCabe-Thiele and the distillate purity move. More reflux, sharper split, more reboiler duty.
4. Move the **feed stage** and re-run — the profiles show whether you've placed it well (a smooth composition front) or badly (a kink and wasted stages).
5. Try pushing the distillate purity past ~89% ethanol — it won't go, because the azeotrope caps it. That's §5.6, live.

A column is just Chapter 3's flash, stacked and wired with reflux and reboil — and everything you've learned so far (thermodynamics, flash, the equation-oriented solve) is doing the work inside that one node.

5.8 Exercises

PRACTICE

Work each problem yourself first, then reveal the solution to check it. Where a problem says so, reproduce it live in MaximaLabs — the solver is the answer key.

1 WARM-UP

Estimate the **minimum number of stages** (total reflux) with the Fenske equation for a separation with relative volatility

$$\alpha = 2.4$$

, a distillate at 95 mol% light key, and bottoms at 5 mol% light key.

Show solution

2 CORE

On a McCabe-Thiele diagram, what happens to the number of stages as the reflux ratio

$$R$$

increases, and what do you pay for it? Name the two limits

$$R \to \infty$$

and

$$R \to R_{\min}$$

.

Show solution

3 CHALLENGE

Push the ethanol-water column toward a 99 mol% ethanol distillate and it stops improving. Explain what the stage profile and McCabe-Thiele diagram are showing, and name three ways to actually get past it.

Show solution

CHAPTER 6

Heat Transfer & Energy Integration

Almost every process spends most of its operating cost on heating and cooling. This chapter is the equipment that moves heat — heaters, coolers, and exchangers — and then the bigger idea that saves the money: making the hot streams heat the cold ones instead of paying for utilities twice.

6.1 Heaters and coolers

The simplest energy units add or remove a duty. You specify either the duty

$$Q$$

or the outlet temperature, and the other follows from the energy balance:

$$Q = F\,(h_{\text{out}} - h_{\text{in}})$$

Q	Duty — heat added (Q>0, a heater) or removed (Q<0, a cooler).
F	Molar flow through the unit.
$h_{\text{in}},\ h_{\text{out}}$	Inlet and outlet molar enthalpy, from the active thermo package.

A heater takes

$$Q > 0$$

, a cooler

$$Q < 0$$

. That's the whole model — but the duty it reports is a real utility cost, and the point of the rest of this chapter is to make it smaller.

WORKED EXAMPLE — SENSIBLE-HEAT DUTY

5 mol/s at

$$C_p = 75\ \mathrm{J/mol{\cdot}K}$$

heated from 300 K to 350 K (the same case as §6.7's core exercise):

$$Q = F\,C_p\,\Delta T = 5 \times 75 \times 50 = 18{,}750\ \mathrm{W} \approx 18.75\ \mathrm{kW}$$

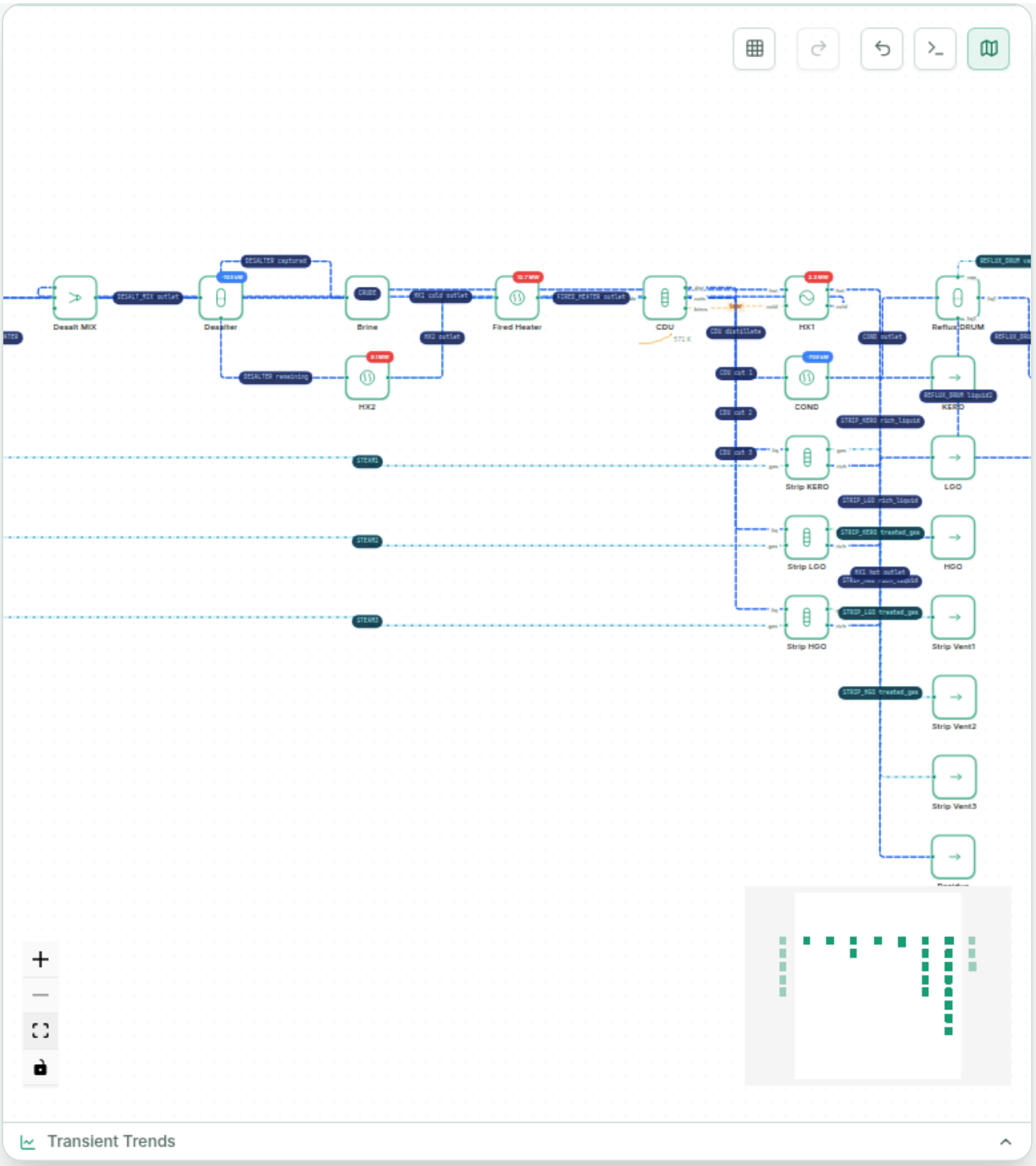
6.2 The heat exchanger

A heat exchanger couples *two* streams: a hot one giving up heat and a cold one taking it. No utility is consumed — the duty simply moves from one side to the other, so its energy balance is a pair of matched terms:

$$Q = F_{\mathrm{hot}}\,(h_{\mathrm{hot,in}} - h_{\mathrm{hot,out}}) = F_{\mathrm{cold}}\,(h_{\mathrm{cold,out}} - h_{\mathrm{cold,in}})$$

F_{hot}, F_{cold}	Molar flow of the hot and cold streams.
$h_{\mathrm{hot,in/out}}$, $h_{\mathrm{cold,in/out}}$	Molar enthalpy of each stream at its inlet and outlet — the duty that leaves the hot side exactly equals the duty that enters the cold side.

The example below is a crude-distillation train where the hot column products preheat the cold crude feed through a feed-effluent exchanger before it ever reaches a fired heater — recovered heat that would otherwise be paid for twice.



A heat-integrated crude train: the feed-effluent heat exchanger recovers heat from hot products into the cold feed before the fired heater — the exchanger consumes no utility, it just relocates duty.

6.3 Rating: how big must the exchanger be?

Knowing the duty isn't the same as knowing the exchanger will fit it. The transfer rate is set by the area

$$A$$

, the overall coefficient

$$U$$

, and the driving temperature difference — the log-mean temperature difference (LMTD):

$$Q = U\,A\,\Delta T_{lm}\,F_T, \qquad \Delta T_{lm} = \frac{\Delta T_1 - \Delta T_2}{\ln(\Delta T_1/\Delta T_2)}$$

U	Overall heat-transfer coefficient — how readily heat crosses the wall between the two fluids.
A	Heat-transfer area.
ΔT_{lm}	Log-mean temperature difference — the effective driving force averaged over the exchanger's length.
$\Delta T_1,\ \Delta T_2$	Temperature difference between the two streams at each end of the exchanger.
F_T	Correction factor (≤1) for a non-pure-counter-current geometry (below).

WORKED EXAMPLE — LMTD

Counter-current terminals: hot 400 K→350 K, cold 300 K→340 K (illustrative):

$$\Delta T_1 = 400 - 340 = 60\ \text{K}, \qquad \Delta T_2 = 350 - 300 = 50\ \text{K}$$

$$\Delta T_{lm} = \frac{60 - 50}{\ln(60/50)} = \frac{10}{0.1823} \approx 54.9\ \text{K}$$

$$\Delta T_{lm}$$

alone only tells the truth for pure counter-current flow. A real shell-and-tube exchanger mixes some co-current pattern in (one shell pass, two-or-more tube passes), so the actual driving force is

$$\Delta T_{lm}$$

knocked down by the **correction factor**

$$F_T \le 1$$

, read off two dimensionless temperature ratios built from the four terminal temperatures (hot in/out

$$T_1, T_2$$

, cold in/out

$$t_1, t_2$$

) — the cold-side thermal effectiveness

$$P$$

and the capacity-rate ratio

$$R$$

:

$$P = \frac{t_2 - t_1}{T_1 - t_1}, \qquad R = \frac{T_1 - T_2}{t_2 - t_1} = \frac{\dot{m}_{\text{cold}} c_{p,\text{cold}}}{\dot{m}_{\text{hot}} c_{p,\text{hot}}}$$

$T_1,\ T_2$	Hot-stream inlet and outlet temperature.
$t_1,\ t_2$	Cold-stream inlet and outlet temperature.
P	Cold-side thermal effectiveness — how much of the maximum possible temperature rise the cold stream actually achieves.
R	Capacity-rate ratio — the hot-to-cold ratio of mass flow times specific heat.

WORKED EXAMPLE — P AND R

Same terminals as the LMTD example above (hot 400→350 K, cold 300→340 K):

$$P = \frac{340 - 300}{400 - 300} = 0.4, \qquad R = \frac{400 - 350}{340 - 300} = 1.25$$

In practice

$$F_T$$

comes from the standard Bowman-Mueller-Nagle charts (built for 1-shell/2-tube-pass geometry, or their closed-form fit) indexed by

$$P$$

and

$$R$$

— MaximaLabs' rating panel below solves it directly rather than reading a chart, and flags the case every design should avoid:

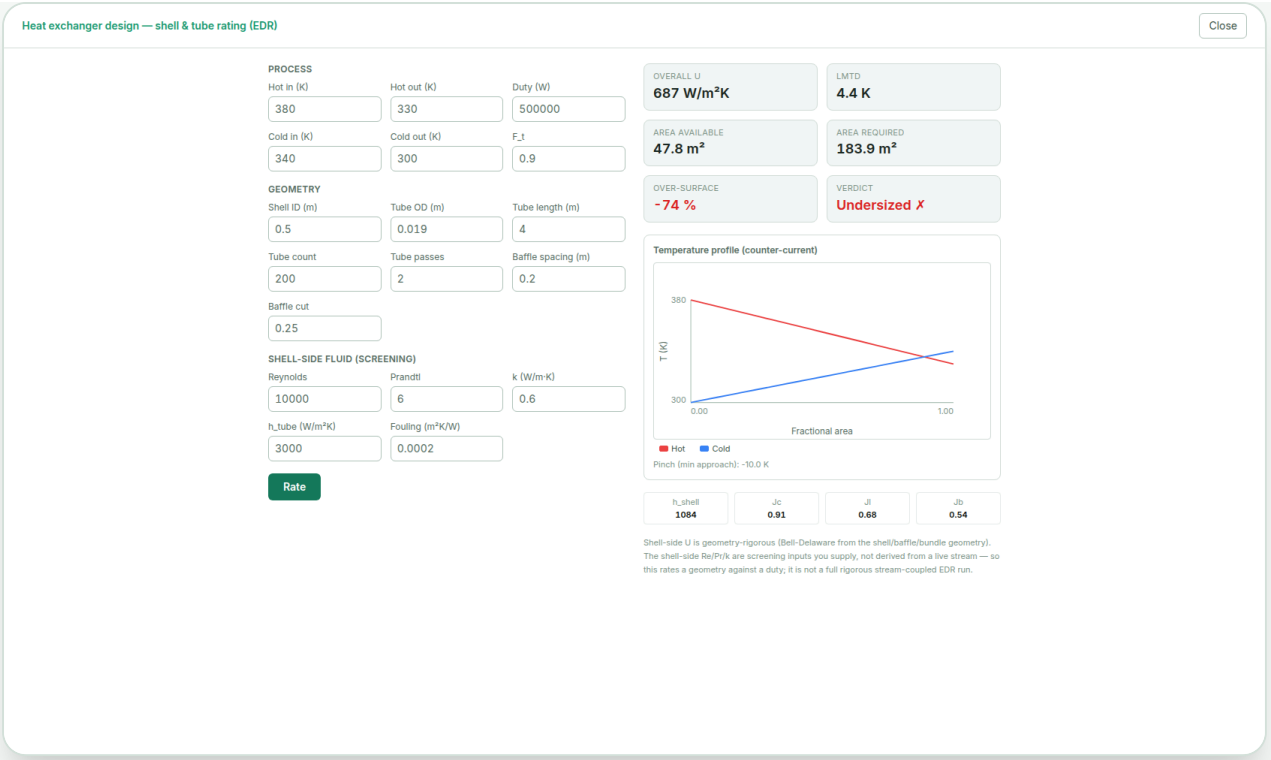
$$F_T < 0.75$$

means the temperature crosses are severe enough that adding a shell pass (or another shell in series) is cheaper than accepting the derated area.

Rating asks: for a given geometry, is the available area enough for the required duty? MaximaLabs' HX design panel (an EDR-style rating) takes the duty, terminal temperatures, and shell/tube/baffle geometry and returns the overall

$$U$$

, the LMTD, the **required vs. available area**, the over-surface margin (adequate or undersized), the Bell-Delaware shell-side factors, and the counter-current temperature profile:



HX design (EDR): rate a geometry against a duty — U, LMTD, required vs. available area, the over-surface margin (✓ adequate / ✕ undersized), and the hot/cold temperature profile down the exchanger. Change geometry and re-rate.

6.4 Energy integration: the pinch

One exchanger recovers heat between two streams. A whole plant has many hot and cold streams, and the question becomes: *what is the least heating and cooling utility this process could possibly need?* That's **pinch analysis**. You combine all hot streams into a hot composite curve and all cold streams into a cold composite curve, slide them together until the closest approach equals a chosen

$$\Delta T_{\min}$$

, and read off the answer:

$$Q_{H,\min},\ Q_{C,\min}\ \text{(minimum utilities)}, \qquad T_{\text{pinch}}\ \text{(the pinch point)}$$

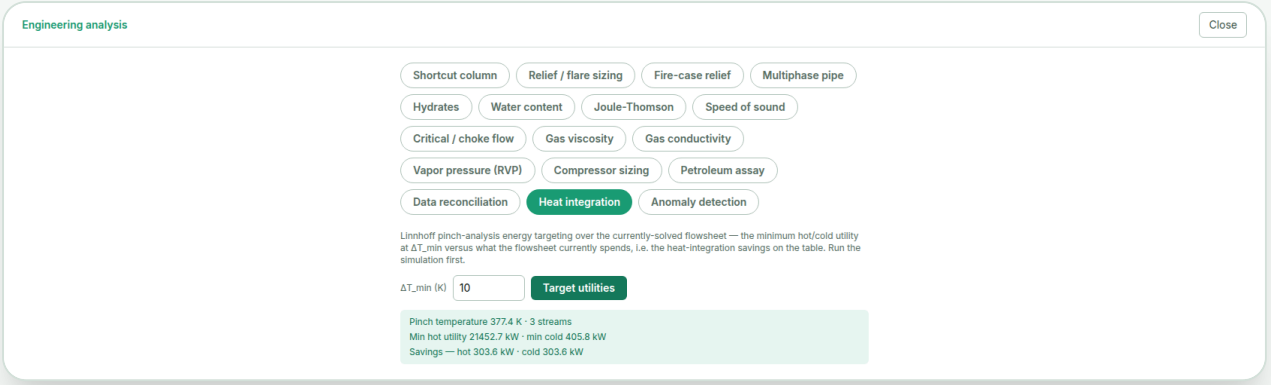
$Q_{H,\min},\ Q_{C,\min}$	Minimum hot and cold utility the process could reach with a perfectly integrated network — the target.
T_{pinch}	The temperature where the hot and cold composite curves come closest (exactly \Delta T_min apart).

WORKED EXAMPLE — PINCH TARGETS

A symmetric two-stream case (hot 100 K→40 K, cold 30 K→90 K, both CP=1 W/K, ΔT_min=20 K) — reproduced from this codebase's own validation test:

$$Q_{H,\min} = Q_{C,\min} = 10\ \mathrm{W}, \qquad T_{\mathrm{pinch}} = 100\ \mathrm{K}$$

The point where the curves touch is the **pinch**; it divides the process into a heat-deficit region above and a heat-surplus region below, and the golden rule — don't transfer heat across the pinch — is what unlocks the minimum-utility target. MaximaLabs runs the Linnhoff targeting over the solved flowsheet and reports the minimum hot/cold utilities and the savings left on the table versus the current design:



Pinch (Linnhoff) targeting over the solved crude train: the pinch temperature, the minimum hot and cold utility the process could use, and the utility savings still available — the number that justifies adding a heat exchanger.

6.5 From target to design: the heat-exchanger network

Targeting (§6.4) tells you the minimum utility a perfectly-integrated network could reach — but not *which* streams to match against which, or how many new exchangers to add. The **pinch design method** answers that with a grid diagram (hot streams left-to- right on top, cold streams below, the pinch drawn as a vertical line splitting the grid) and one governing rule: split the problem at the pinch and design each side separately, since no stream may legally cross it.

Immediately next to the pinch, a match is only **feasible** if it can carry the full duty without a temperature cross. That reduces to a heat-capacity-flow (

$$CP = \dot{m}\, c_p$$

) rule, different on each side:

$$\text{above the pinch: } CP_{\text{hot}} \le CP_{\text{cold}}, \qquad \text{below the pinch: } CP_{\text{hot}} \ge CP_{\text{cold}}$$

CP	Heat-capacity flow rate \dot m c_p — how much duty a stream gives up (or absorbs) per degree.
CP_{hot}, CP_{cold}	The CP of the hot and cold stream in a candidate match — which one must be larger sets which side of the pinch the match is feasible on.

The **tick-off heuristic** then greedily matches streams pinch-outward: at each step, match the largest feasible pair, exhaust ("tick off") whichever stream runs out of duty first, and carry the other's remainder to the next match. Repeat until every stream is either fully matched against another process stream or left with a residual duty — which becomes the hot/cold utility this design actually needs. MaximaLabs runs exactly this algorithm over your solved flowsheet's real hot and cold streams and returns the match list, each exchanger's estimated area, and — honestly — a flag on any stream a stream-split would be needed to match (the one case tick-off alone can't resolve): open the same Energy panel's **Network design** tab (right beside the targeting view above) and it lists the proposed matches, each one's duty and estimated area, and the utility remaining after matching — the same streams §6.4 targeted, now actually assigned to exchangers.

This is screening-level, not detailed exchanger design — no stream-split optimization, no capital-cost weighting between candidate networks — but it's the real next step after a target: an actual list of exchangers to add, sized well enough to cost and specify.

6.6 Try it

REPRODUCE IT IN YOUR BROWSER

1. Open a heat-integrated example — the crude distillation train — and **Run** it.
2. Click the feed-effluent **heat exchanger** and read its two-sided duty and the hot/cold terminal temperatures in the stream table.
3. Open **HX design (EDR)** (right rail ▸ Analysis & reports), hit **Rate**, and read the LMTD, U, and required-vs-available area. Shrink the shell diameter and re-rate to watch the over-surface margin go negative (undersized).
4. Open **Engineering analysis ▸ Heat integration**, set a

$$\Delta T_{\min}$$

, and click **Target utilities** to see the minimum-utility target and the savings against the current design.

5. Switch to the **Network design** tab — same panel — to see the tick-off match list that actually reaches the target: which streams pair up, and what's left as genuine hot/cold utility.

Heat transfer is where a correct simulation turns into a cheaper plant: size the exchangers so they fit, then integrate them so the process heats itself. Next: turning a converged model into decisions — sensitivity, design specs, and optimization.

6.7 Exercises

PRACTICE

Work each problem yourself first, then reveal the solution to check it. Where a problem says so, reproduce it live in MaximaLabs — the solver is the answer key.

1 WARM-UP

A counter-current exchanger cools a hot stream from 150 °C to 90 °C while heating a cold stream from 40 °C to 110 °C. Compute the log-mean temperature difference.

Show solution

2 CORE

How much duty is needed to heat 5 mol/s of a stream with molar heat capacity

$$C_p = 75\ \mathrm{J/mol{\cdot}K}$$

from 300 K to 350 K (no phase change)?

Show solution

3

CHALLENGE

In pinch analysis, what is the **pinch**, and what are the three golden rules for reaching the minimum-utility target? What does violating them cost? Check the targets on the crude distillation train heat-integration panel.

Show solution

CHAPTER 7

Process Analysis & Optimization

A converged flowsheet answers one question: *what happens at these conditions?* Real engineering asks better ones — how sensitive is the answer, what input hits a target, which design is cheapest. This chapter turns the model from a calculator into a decision tool.

7.1 Sensitivity analysis

The first question is always "what if?" A **sensitivity sweep** varies one input

$$x$$

across a range and re-solves the whole flowsheet at each point, tracing an output

$$y = f(x)$$

:

$$y_k = f(x_k), \qquad x_k \in [x_{\min},\, x_{\max}], \quad k = 1 \dots N$$

x_k	The k-th value of the swept input (e.g. reflux ratio) — chosen at N evenly spaced points across the range.
$x_{\min},\ x_{\max}$	The lower and upper bound of the sweep range you set.
N	Number of sweep points — each one a full, independent re-solve of the flowsheet.
y_k	The tracked output (e.g. reboiler duty) at that point — the solver's real answer, not an interpolation.

For the ethanol-water column, sweeping the reflux ratio against the reboiler duty shows the central trade-off of distillation: more reflux sharpens the separation but costs energy, and the curve tells you where the knee is. MaximaLabs runs each point as a real re-solve (an enqueued job) and plots the response:

A sensitivity sweep of reflux ratio vs. reboiler duty for the ethanol-water column — every point is a full flowsheet re-solve. This is the reflux/energy trade-off, drawn.

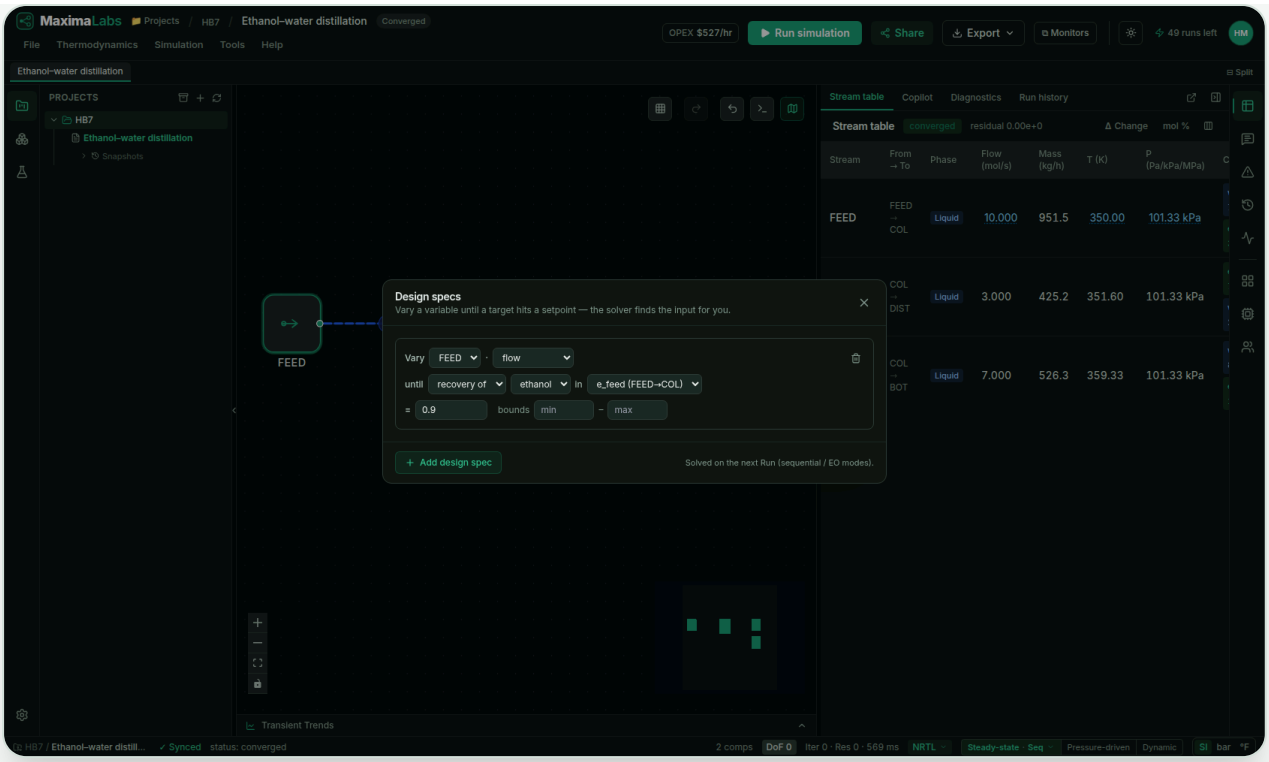
7.2 Design specs: state the result, solve for the input

Sensitivity answers "what does this input do?" A **design spec** answers the inverse — "what input gives me this result?" You name a manipulated variable and a target, and the solver root-finds the input that makes the target hit its setpoint:

$$\text{find } x \;\text{ such that }\; g(x) = s^* \qquad (\text{e.g. vary reflux until distillate purity} = 0.8)$$

x	The manipulated variable — a single flowsheet parameter you designate (e.g. reflux ratio).
$g(x)$	The target quantity as a function of that input (e.g. distillate ethanol mole fraction) — computed by a full flowsheet re-solve at each trial x.
s^*	The setpoint you want the target to hit (e.g. 0.8).

Instead of guessing reflux ratios until the purity comes out right, you state the purity and MaximaLabs finds the reflux — the same "Design Spec" idea as Aspen, or a SimCentral *adjust*. Each spec reads as a sentence and reports whether it was met on the next Run:



The Design Spec panel: 'vary [node · param] until [recovery/purity of …] = [setpoint]'. The solver finds the input for you and reports met / not-met with the solved value on the next Run.

7.3 Case studies

A sensitivity sweep moves one variable continuously; a **case study** instead evaluates a small table of *named, discrete* scenarios — winter vs. summer feed, three catalyst activities, a set of turndown rates — each one a full set of

$$\{\text{node}, \text{param}, \text{value}\}$$

overrides, re-solved from scratch and laid side by side against whichever outputs you care about:

$$y_{c,o} = f_o(x_c), \qquad c = 1 \dots C \ (\text{named cases}), \quad o = 1 \dots O \ (\text{tracked outputs})$$

x_c	The full set of {node, param, value} overrides that define named case c (e.g. Winter feed's colder, wetter composition).
C	Number of named cases (here 3: Design, Winter feed, Turndown) — discrete scenarios you actually need answers for, not a swept continuum.
f_o	The flowsheet re-solve, read at tracked output o (e.g. distillate purity, reboiler duty).

O	Number of tracked outputs (here 2: distillate purity and reboiler duty).
$y_{c,o}$	The solved value of output o under case c — one cell of the results grid below.

Take the ethanol-water column and ask "what does the winter feed do to us?" — define **Design** (the current feed/reflux), **Winter feed** (colder, wetter feed composition), and **Turndown** (70% of design feed rate) as three cases, track distillate purity and reboiler duty as outputs, and MaximaLabs re-solves all three and returns one grid — no sweep range to define, because these are three specific, named operating points you actually need answers for, not a continuum:

Case	Distillate purity	Reboiler duty
Design	0.80 mol frac.	1,020 kW
Winter feed	0.77 mol frac.	1,140 kW
Turndown (70%)	0.83 mol frac.	720 kW

The pattern each row shares: colder/wetter feed costs both purity *and* energy (more water to reboil off), while turndown trades duty for purity in the other direction — reading that off a grid, not three separate manual runs, is the entire point of a case study. It lives on the same Plots panel as the sweep (§7.1), and reuses the identical re-solve mechanics — a case study is a sensitivity sweep over discrete named points instead of a continuous range. For four full worked case studies — reaction, separation, and economics chained across whole flowsheets, not just one column's named scenarios — see Chapter 11.

7.4 Optimization

Sensitivity and case studies *explore*; optimization *decides*. Before you can even pose the problem, you need to know whether there's a decision to make at all — the same **degrees-of-freedom** count from Chapter 1, applied to the whole optimization problem rather than one flowsheet solve:

$$N_f = (\text{independent process variables}) - (\text{independent equations})$$

N_f	Degrees of freedom of the optimization problem — the count of variables genuinely free to be decided.

$\text{independent process variables}$ Every unknown quantity in the problem (flows, temperatures, pressures, compositions, equipment parameters) not already fixed by a spec.

$\text{independent equations}$ Every material/energy balance and equilibrium relation that must hold — each one removes one degree of freedom.

WORKED EXAMPLE — A TINY ILLUSTRATIVE COUNT

Take a small piece with **3 process variables** (say, a feed flow, a feed temperature, and a duty) and **2 independent equations** relating them (a material balance and an energy balance):

$$N_f = 3 - 2 = 1$$

One genuine degree of freedom remains — one decision to make (e.g. the duty), and the other two variables follow from the balances once it's chosen. This is deliberately just arithmetic to show the mechanics of the count, not a solved MaximaLabs case; the same bookkeeping applied to a real column (see Chapter 1's feed→heater→reactor degrees-of-freedom walkthrough) is what tells you whether "optimize this" even makes sense before you pose an objective.

Three cases fall out of the sign of

$$N_f$$

: at

$$N_f = 0$$

the problem is *exactly specified* — there's nothing left to optimize, only a system to solve (this is every ordinary flowsheet Run in this Handbook so far); at

$$N_f < 0$$

it's *over-specified* — too many constraints for the variables you have, no feasible solution exists until you relax one; only at

$$N_f > 0$$

is the problem *under-specified* in the useful sense — genuinely free variables exist, and picking their best values is what optimization means. With real freedom established, you give an objective and let a solver search those free decision variables for the best feasible point:

$$\min_{x} \; f(x) \quad \text{subject to} \quad g(x) = 0, \;\; h(x) \le 0, \;\; x^{L} \le x \le x^{U}$$

x	The decision variables — the genuinely free variables N_f identified above (e.g. reflux ratio, feed preheat).
$f(x)$	The objective function being minimized (installed cost, energy, carbon — whichever you choose).
$g(x) = 0$	Equality constraints — the flowsheet's own material and energy balances, which must always hold exactly.
$h(x) \le 0$	Inequality constraints — product specs and equipment limits the solution must not violate (e.g. purity ≥ 95%).
$x^{L},\ x^{U}$	Lower and upper bounds on each decision variable — the physically/operationally sane range the search stays inside.

The objective

$$f$$

might be installed cost, energy, or carbon; the equality constraints

$$g$$

are the flowsheet's own balances; the inequalities

$$h$$

are your product specs and equipment limits. MaximaLabs drives this with a real optimizer (Pyomo/SciPy) rather than an LLM guessing — the AI copilot can *propose* what to optimize and set it up, but the deterministic solver finds every number. The built-in objectives (lowest installed cost, lowest carbon footprint) are one click from the Simulation ▸ Optimizer menu; a custom objective and decision-variable set is a short spec away.

7.5 Try it

REPRODUCE IT IN YOUR BROWSER

1. Open the ethanol-water distillation example and **Run** it.

2. Open **Plots ▸ Sensitivity** (right rail ▸ Analysis & reports), vary the column's **reflux_ratio** against the **reboiler_duty**, and **Run sweep** to draw the energy trade-off.
3. Open **Simulation ▸ Design specs…**, add a spec that varies reflux until the distillate ethanol purity hits a setpoint, and **Run** — read the solved reflux back from the met/not-met badge.
4. Open **Plots ▸ Case study**, define two or three named scenarios (e.g. a colder feed temperature, a lower feed rate), track distillate purity and reboiler duty as outputs, and **Run** to see all cases solved side by side in one grid.
5. Open **Simulation ▸ Optimizer ▸ Optimize for lowest installed cost** and watch the solver pick the decision variables — then compare its choice to the knee you saw in the sweep.

Sweep to understand, spec to hit a target, optimize to decide — all on the same converged model from the earlier chapters. Last stop: adding time, with dynamic simulation.

7.6 Exercises

PRACTICE

Work each problem yourself first, then reveal the solution to check it. Where a problem says so, reproduce it live in MaximaLabs — the solver is the answer key.

1. WARM-UP

   You want the ethanol-water column's distillate to hit exactly 95 mol% ethanol. Set this up as a **design spec**: what is the manipulated variable, and what does the solver actually do?

   Show solution

2. CORE

   You sweep the reflux ratio from 1.5 to 5 and plot both distillate purity and reboiler duty. What shape does each curve take, and what is the engineering takeaway?

   Show solution

3

CHALLENGE

Now **minimize reboiler duty subject to distillate ≥ 95 mol%**. Why is the purity constraint *active* (binding) at the optimum, and what does the optimizer move? Reproduce it in the Optimization studio on the ethanol-water column.

Show solution

CHAPTER 8

Dynamic Simulation

Every chapter until now solved for a single steady state — the plant sitting still. But plants start up, shut down, get disturbed, and are held in place by controllers. Dynamic simulation adds the one thing steady state leaves out: **time**.

8.1 From steady state to time

A steady-state balance says what goes in equals what goes out. A dynamic balance keeps the term steady state throws away — **accumulation**. For the holdup of component

$$i$$

in a unit:

$$\frac{dM_i}{dt} = \sum_{\text{in}} \dot{n}_i - \sum_{\text{out}} \dot{n}_i + \nu_i \, r$$

M_i	Holdup of component i in the unit (mol) — the new state variable dynamic simulation adds on top of steady state.
$\dot{n}_i$	Molar flow rate of component i (mol/s) in a given inlet or outlet stream.
ν_i	Stoichiometric coefficient of i in the reaction (positive for a product, negative for a reactant, zero if i doesn't react).
r	Reaction rate (mol/s) — zero for a purely physical (non-reacting) holdup, e.g. a tank or column sump.

Now the model is a set of **differential equations** (one per holdup) coupled to the same **algebraic equations** as before (equilibrium, summation, energy) — a **DAE** system, integrated forward in time by an implicit solver. Set every

$$dM_i/dt = 0$$

and you're back to Chapter 1's steady state; that's not a coincidence, it's the same model with the clock stopped.

WORKED EXAMPLE — A WELL-MIXED TANK'S FIRST-ORDER RESPONSE

Take a single holdup with no reaction:

$$V\,dC/dt = F\,(C_{\text{in}} - C)$$

, a first-order lag with time constant

$$\tau = V/F$$

. Say

$$\tau = 10$$

minutes and the inlet concentration steps to a new value. By hand:

$$\text{at } t = \tau: \quad 1 - e^{-1} = 1 - 0.3679 = 0.6321 \;\; (63.2\% \text{ of the change})$$

$$\text{at } t = 4\tau: \quad 1 - e^{-4} = 1 - 0.0183 = 0.9817 \;\; (98.2\% \text{ of the change})$$

Hence the standard rule of thumb: a first-order holdup covers ~63% of a step change in one time constant and is effectively settled in about four — the same check applies to any tank level, column sump composition, or HX metal temperature the dynamic solver tracks.

8.2 Initialized from steady state

A dynamic run needs a consistent starting point, and the natural one is the converged steady-state solution you already have. MaximaLabs runs the dynamics *from the steady-state design* — no re-authoring the flowsheet — so time zero is the answer from the earlier chapters, and you watch the plant move away from it and settle back.

8.3 Control loops

The reason most plants are dynamic-but-stable is **control**. A feedback controller measures a process variable (PV), compares it to a setpoint (SP), and moves a manipulated variable to close the error

$$e = \mathrm{SP} - \mathrm{PV}$$

. The workhorse is the PID law:

$$u(t) = K_c \left[e(t) + \frac{1}{\tau_I} \int_0^t e(t')\, dt' + \tau_D \frac{de}{dt} \right]$$

$u(t)$	Controller output — the manipulated-variable signal sent to the final control element (e.g. a valve position or a heater duty).
K_c	Controller gain — how strongly the output reacts to a given error (proportional action).
$e(t)$	Error at time t, e = SP − PV (setpoint minus the measured process variable).
τ_I	Integral time — smaller values make integral action stronger, accumulating past error faster.
τ_D	Derivative time — larger values react more strongly to how fast the error is currently changing.

WORKED EXAMPLE — WHY PROPORTIONAL-ONLY CONTROL LEAVES AN OFFSET

A tank temperature loop: setpoint 350 K, measured (PV) 345 K, so

$$e = \mathrm{SP} - \mathrm{PV} = 350 - 345 = 5\ \mathrm{K}$$

. With a P-only controller (

$$\tau_I \to \infty$$

, no derivative) and

$$K_c = 2$$

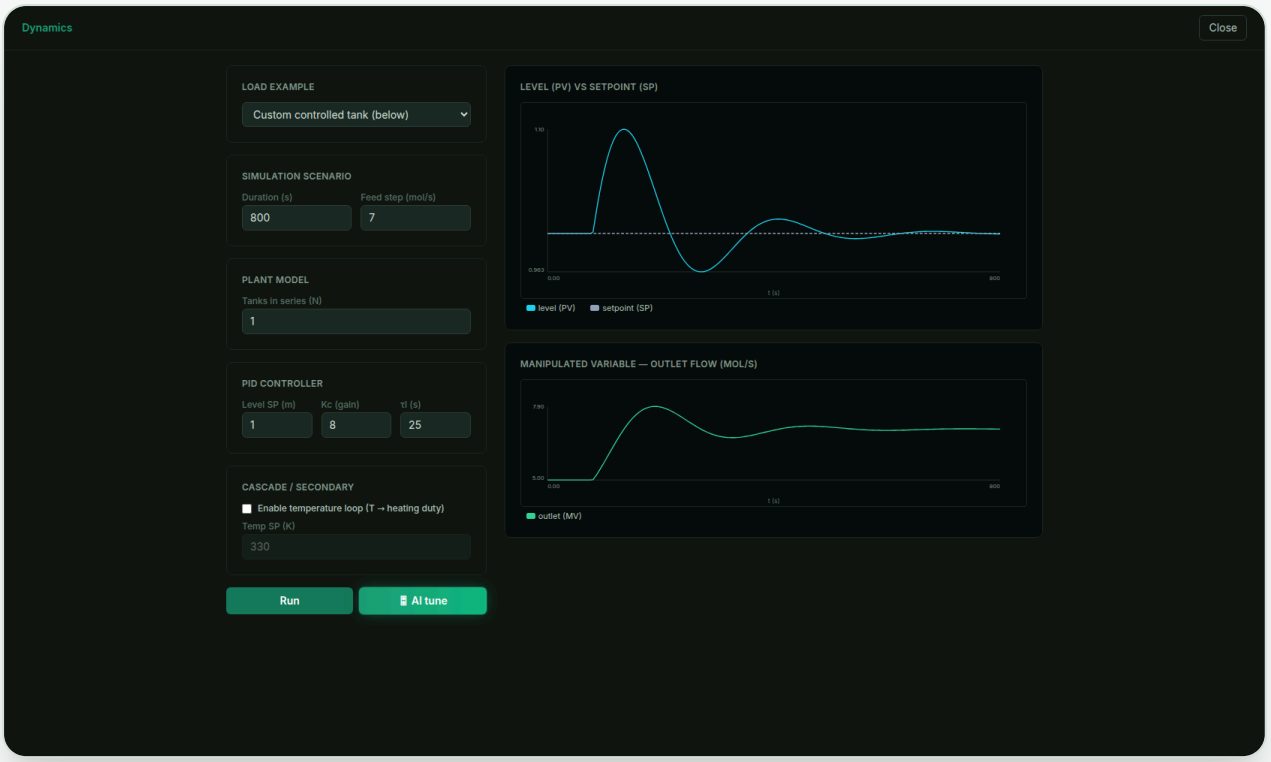
, the proportional term alone contributes

$$K_c\, e = 2 \times 5 = 10$$

to the output.

To keep feeding that +10 into the final control element — which is exactly what's needed to counteract a steady load — the controller must keep seeing a nonzero error. If the error ever reached zero, the output would drop back to just its bias and the load would pull the temperature away from setpoint again. So the loop settles with a permanent 5 K **offset**; only adding the integral term removes it (§8.6, exercise 2).

Below, a level controller on a tank is disturbed by a feed-flow step and pulls the level (PV) back to its setpoint (SP) — the classic closed-loop response. Tune the gains and you trade off speed against overshoot, exactly as on a real DCS:



A PID level loop rejecting a feed-flow disturbance: the level (PV) departs from setpoint (SP) when the feed steps, then the controller drives it back. Tuning the gains trades speed for overshoot.

8.4 Running a whole flowsheet in time

Dynamics isn't only for single tanks. Switch the status-bar solve mode to **Dynamic** and MaximaLabs compiles *any* steady-state flowsheet — columns, reactors, recycle loops and all — into a DAE and integrates it from its steady-state design (no re-authoring). Pick a holdup and it charts that unit's composition or temperature vs. time; add a feed step (a flow, temperature, or composition change at a chosen instant) and you watch the disturbance propagate through the plant and settle to a new steady state.

This is the payoff of the *unified model*: the steady-state flowsheet, the property methods, the reactors and columns you built in the earlier chapters are exactly what runs dynamically — you add time, not a second model.

8.5 Try it

REPRODUCE IT IN YOUR BROWSER

1. Open any converged example (e.g. the ethanol-water column) and **Run** it to steady state first.
2. Open **Simulation ▸ Dynamic simulation…** for the controlled-tank scenario, set a level setpoint and a feed-flow disturbance, and **Run** to watch the PID loop pull the level back.
3. Retune the controller gains (K_c , τ_I) and re-run — see the response get faster but start to overshoot.
4. Switch the status-bar solve mode to **Dynamic** to run the whole flowsheet in time and chart each holdup's transient toward steady state.

And that's the arc: build a flowsheet as a graph, pick the thermodynamics, flash it, react it, separate it, integrate its heat, analyze and optimize it — then add time. Every step used the same model, and every number came from a deterministic solver you could see the math behind. That's the whole of process simulation, and you did it in a browser.

8.6 Exercises

PRACTICE

Work each problem yourself first, then reveal the solution to check it. Where a problem says so, reproduce it live in MaximaLabs — the solver is the answer key.

1 **WARM-UP**

A well-mixed tank of constant liquid volume V has an inlet flow F

at concentration

$$C_{\text{in}}$$

and an equal outlet flow

$$F$$

at concentration

$$C$$

. Write the dynamic component balance and give the time constant.

Show solution

2 CORE

A **proportional-only** controller leaves a steady-state **offset** from setpoint. Why? And how does adding **integral** action remove it?

Show solution

3 CHALLENGE

A step increase in feed flow disturbs a temperature loop under PID control. Describe the transient the controlled temperature follows, and how the tuning sets its shape. Reproduce a disturbance in a live loop with the Dynamics / live-PID panel.

Show solution

CHAPTER 9

Fluid Handling Equipment

Every flowsheet so far has quietly assumed streams arrive at the pressure a unit op needs. In a real plant, getting them there is its own equipment class: pumps and compressors raise pressure, turbines and valves drop it, and pipe itself eats pressure to friction over distance. None of this changes composition — it's pure mechanical/thermal energy balance — but getting it wrong is a common reason a "correct" flowsheet doesn't match the real plant.

9.1 Pumps: raising liquid pressure

A pump does mechanical work on an (incompressible) liquid. The theoretical work is flow times the pressure rise divided by density; a real pump needs more than that, by the **hydraulic efficiency**

$$\eta$$

:

$$W = \frac{F\,(P_{out} - P_{in})}{\rho\,\eta}$$

W	Shaft (pump) work delivered to the fluid, watts (J/s).
F	Molar flow through the pump, mol/s.
$P_{out},\ P_{in}$	Outlet and inlet pressure, Pa.
ρ	Molar density of the liquid at inlet conditions, mol/m³.
η	Hydraulic efficiency (0–1) — the fraction of shaft work that goes into raising pressure rather than being lost as heat.

$$F$$

is the molar flow,

$$\rho$$

the liquid density at the inlet conditions. The shaft work

$$W$$

shows up as sensible heating of the liquid too (the inefficiency has to go somewhere) — small for water, but not always negligible for viscous or high-head service. A real pump's head isn't flat with flow either: MaximaLabs' pump can take a

$$\Delta P = \rho g \, H(Q)$$

head curve instead of a fixed outlet pressure, so raising throughput on a pinned curve genuinely reduces the achievable head — the same curve a vendor datasheet gives you.

WORKED EXAMPLE — SAME CASE AS §9.6'S WARM-UP EXERCISE

5 mol/s of water (molar density

$$\rho \approx 55{,}390 \ \mathrm{mol/m^3}$$

) pumped from 1 atm to 10 atm (

$$\Delta P \approx 9.12 \times 10^5 \ \mathrm{Pa}$$

) at 70% hydraulic efficiency:

$$W = \frac{5 \times 9.12 \times 10^5}{55{,}390 \times 0.70} \approx 118 \ \mathrm{W}$$

The same arithmetic §9.6's exercise walks through in full — restated here next to the equation it comes from. 118 W is a modest duty: pumping liquids is cheap, which is why §9.2's compressor example below (compressing a gas, not a liquid) needs so much more power for a comparable pressure ratio.

9.2 Compressors and turbines: isentropic + efficiency

Compressible fluids need the real thermodynamics from Chapter 2, not just a pressure/density ratio. MaximaLabs solves an **ideal (isentropic) path** first — find the outlet temperature

$$T_{2s}$$

that keeps entropy constant across the pressure change, using whatever equation of state or activity model the flowsheet is running:

$$S(T_{2s}, P_2) = S(T_1, P_1) \quad \text{(solved for } T_{2s}\text{)}$$

$T_1,\ P_1$	Known inlet temperature and pressure.
P_2	Known outlet pressure (the compression/expansion target).
T_{2s}	Ideal (isentropic) outlet temperature — the one unknown, solved so entropy is unchanged across the pressure step.
$S(T, P)$	Molar entropy at (T,P) from the flowsheet's active thermo package — an equation of state or activity model, per Chapter 2, not an ideal-gas shortcut.

then applies the isentropic efficiency

$$\eta$$

to get the *real* outlet state and the actual shaft work — for a compressor the real temperature rise is bigger than the ideal one (efficiency losses show up as extra heat):

$$T_2 = T_1 + \frac{T_{2s} - T_1}{\eta}, \qquad W = F\left(H(T_2, P_2) - H(T_1, P_1)\right)$$

T_2	Real (actual) outlet temperature — always hotter than the ideal T_{2s} for a compressor, since inefficiency adds extra heating (see §9.6's core exercise).
η	Isentropic efficiency (0–1): the ratio of ideal to actual work for a compressor.
W	Real shaft work delivered to the gas, watts.
F	Molar flow through the machine, mol/s.
$H(T, P)$	Molar enthalpy at (T,P) from the active thermo package.

WORKED EXAMPLE — ISENTROPIC COMPRESSION OF CO2, PR PACKAGE

A small illustrative case computed directly from MaximaLabs' Peng-Robinson package (not from a saved showcase example — a standalone solver call, so it's genuinely computed but not the same run as the co2-capture-compression Try-it case below): pure CO_2 at

$$T_1 = 313.15\ \mathrm{K}$$

(40 °C),

$$P_1 = 1\ \mathrm{bar}$$

, compressed to

$$P_2 = 8\ \mathrm{bar}$$

at 78% isentropic efficiency.

$$S(T_{2s}, P_2) = S(T_1, P_1) \ \Rightarrow\ T_{2s} \approx 476.9\ \mathrm{K}$$

$$T_2 = 313.15 + \frac{476.9 - 313.15}{0.78} \approx 523.1\ \mathrm{K}$$

$$W/F = H(T_2, P_2) - H(T_1, P_1) \approx 8{,}740\ \mathrm{J/mol} \ \Rightarrow\ W \approx 8.74\ \mathrm{kW\ per\ mol/s}$$

Compare to the pump example above: raising a gas's pressure 8× costs roughly two orders of magnitude more work per mole than raising a liquid's pressure 10× — exactly why compressors, not pumps, dominate a plant's utility bill.

A turbine runs the same isentropic solve in reverse (dropping pressure) and applies

$$\eta$$

the other way — the real outlet temperature ends up *higher* than the ideal expansion, because inefficiency means less energy actually left the fluid as shaft work:

$$T_2 = T_1 + \eta\,(T_{2s} - T_1), \qquad W_{shaft} = F\,\big(H(T_1, P_1) - H(T_2, P_2)\big)$$

T_2	Real (actual) outlet temperature — always warmer than the ideal T_{2s} for a turbine too, since inefficiency means less energy leaves as work.
η	Isentropic efficiency (0–1): the ratio of actual to ideal work extracted.

W_{shaft}	Real shaft work extracted from the fluid, watts.

Because both use the flowsheet's real

$$H(T, P)$$

/

$$S(T, P)$$

from the active thermo package (Chapter 2) rather than an ideal-gas

$$\gamma$$

shortcut, a compressor on a non-ideal gas — dense-phase CO_2, a real natural-gas mix — gets a genuinely different answer than the textbook ideal-gas formula would. A large pressure ratio in one step is unrealistic for real machinery; MaximaLabs' **multistage compressor**splits it into

$$n$$

stages of equal pressure ratio

$$r = (P_{out}/P_{in})^{1/n}$$

, each solved by the same isentropic-efficiency method above, with intercooling between stages — the same reason real plants stage compression instead of doing it in one jump.

9.3 Valves: isenthalpic throttling

A control valve drops pressure with no shaft work and (to a good approximation) no heat loss — a **Joule-Thomson** throttle. The energy balance collapses to a single statement: enthalpy is conserved across the valve, and the outlet temperature is whatever that enthalpy implies at the new, lower pressure:

$$h_{out} = h_{in} \ (\text{isenthalpic}) \ \Rightarrow \ T_{out} = T(P_{out}, h_{in})$$

h_{in}	Inlet molar enthalpy — known from the upstream stream's (T,P).

h_{out}	Outlet molar enthalpy — equal to h_{in}, since a valve does no shaft work and (ideally) exchanges no heat.
P_{out}	Known outlet (downstream) pressure — the valve's set drop.
T_{out}	Outlet temperature — the unknown, solved as the (P,H) flash of §3.6 at the fixed (P_{out}, h_{in}).

For an ideal gas that's no temperature change at all; for a real gas or a liquid near its bubble point it can mean significant cooling — flashing part of the stream to vapor, or (for a dense-phase fluid like the CO_2 in §9.5) dropping across a phase boundary entirely. The isenthalpic solve is exactly why a J-T valve shows up as a cheap way to get refrigeration rather than just "friction that wastes pressure."

9.4 Pipelines: friction, not equipment

Pipe itself isn't a piece of "equipment" in the traditional sense, but it drops pressure the same way a valve does — and unlike a valve, that drop is unavoidable and grows with distance. The classic **Darcy-Weisbach** equation:

$$\Delta P = f\,\frac{L}{D}\,\frac{\rho v^2}{2} + \rho g L \sin\theta, \qquad f = f(\mathrm{Re},\, \varepsilon/D)$$

ΔP	Total pressure drop along the pipe segment, Pa.
f	Darcy friction factor (dimensionless) — a function of Reynolds number and relative roughness.
$L,\ D$	Pipe length and internal diameter, m.
$\rho,\ v$	Fluid mass density (kg/m³) and mean velocity (m/s).
$g,\ \theta$	Gravitational acceleration and the pipe's inclination angle from horizontal — the elevation term vanishes for a flat run (\theta=0).
$\mathrm{Re},\ \varepsilon/D$	Reynolds number and relative pipe roughness (roughness height over diameter) — the two things f depends on.

splits the loss into **friction** (Darcy friction factor

$$f$$

, a function of Reynolds number and relative pipe roughness

$$\varepsilon/D$$

) and **elevation** (the

$$\sin\theta$$

term, zero for a flat run). Over a 150 km trunk line the friction term alone can eat megapascals — which is exactly why long-distance pipeline transport needs intermediate pump/compressor stations to restore pressure, not because the fluid "runs out" of anything.

9.5 Try it

REPRODUCE IT IN YOUR BROWSER

1. Open the CO_2 capture + compression example and **Run** it: an absorber recovers 90% of the flue CO_2, a compressor raises it to 8 atm, and an after-cooler brings it back to 313 K.
2. Click the **COMP** node and open the **Theory** tab — the exact isentropic-efficiency equations from §9.2, with the solved

   $$T_1$$

   ,

   $$T_2$$

   , and shaft work filled in for this stream.
3. Lower the compressor **efficiency** and re-run — watch the outlet temperature and shaft work both rise for the same pressure ratio, exactly as §9.2 predicts.
4. Open the dense-phase CO_2 pipeline example: two 150 km Darcy-Weisbach segments with a pump station (**BOOST**) in between restoring the pressure the first segment's friction cost.

9.6 Exercises

PRACTICE

Work each problem yourself first, then reveal the solution to check it. Where a problem says so, reproduce it live in MaximaLabs — the solver is the answer key.

1 WARM-UP

A pump raises 5 mol/s of water (density 997 kg/m³, molar mass 0.018 kg/mol) from 1 atm (101,325 Pa) to 10 atm at 70% hydraulic efficiency. Estimate the shaft work.

Show solution

2 CORE

Explain why a real compressor's outlet temperature is *higher* than the ideal isentropic outlet temperature, while a real turbine's outlet temperature is also *higher* than its ideal isentropic outlet temperature (not lower, in both cases) — using the

$$\eta$$

equations of §9.2.

Show solution

3 CHALLENGE

In the CO_2 pipeline example, the fluid enters as a liquid-like dense phase at 15 MPa. If the booster pump under-delivers and the pressure after the second 150 km segment drops below CO_2's critical pressure (7.38 MPa) while still near 305 K, what happens to the fluid, and why does that matter operationally? (Hint: revisit the phase diagram from Chapter 2.)

Show solution

CHAPTER 10

Solids Operations

Every unit op so far has treated a stream as a single homogeneous fluid. The moment a solid forms — crystals in a leach liquor, catalyst fines, a precipitated hydroxide cake — a whole second physics kicks in: particles have a *size*, and how big they are decides how fast you can filter them, how wet the resulting cake stays, and how much heat it takes to dry it. This chapter follows one crystal from the mother liquor to a dry powder.

10.1 Crystallization: nucleation, growth, and the MSMPR

A crystal doesn't appear at a fixed size — a population of them nucleates and grows together. The workhorse model is the **MSMPR** (mixed-suspension, mixed-product-removal) crystallizer: nucleation rate

$$B$$

and linear growth rate

$$G$$

both driven by the same supersaturation

$$\Delta C = C - C_{sat}$$

, each with its own empirical order:

$$B = k_b \, \Delta C^{b}, \qquad G = k_g \, \Delta C^{g}$$

B	Nucleation rate — number of new crystals born per unit volume per unit time (#/m³·s).
G	Linear growth rate — how fast a crystal's characteristic length grows (m/s).

$k_b,\ k_g$	Nucleation and growth rate constants — fitted per system, not universal (from lab crystallization data).
$b,\ g$	Empirical orders of the supersaturation dependence — typically b > g, which is why raising supersaturation nucleates faster than it grows (more, smaller crystals).
ΔC	Supersaturation — the actual solute concentration C above the saturation concentration C_{sat} at the operating temperature; the driving force for both B and G.

Rather than track every crystal size individually, the MSMPR model works in **moments** of the size distribution —

$$m_0$$

(crystal count),

$$m_1$$

(total length),

$$m_2$$

(area-related),

$$m_3$$

(volume, i.e. mass) — each moment built from the one below it over the residence time

$$\tau$$

:

$$m_0 = B\tau, \quad m_1 = m_0 G\tau, \quad m_2 = 2m_1 G\tau, \quad m_3 = 3m_2 G\tau, \qquad \bar{L} = m_1/m_0 = G\tau$$

$m_0,\ m_1,\ m_2,\ m_3$	The zeroth through third moments of the crystal-size distribution — number, total length, area-related, and volume/mass-related, each built from the one below it via the growth rate G over the residence time τ.
τ	Residence time — how long a crystal spends in the well-mixed crystallizer volume before leaving with the product stream (s).
$\bar{L}$	Mean crystal size — the population-average characteristic length, m_1/m_0, which collapses to Gτ for the MSMPR's simple moment

cascade.

WORKED EXAMPLE — MEAN CRYSTAL SIZE

A crystallizer growth rate

$$G = 2 \times 10^{-8}$$

m/s and residence time

$$\tau = 3600$$

s give mean crystal size:

$$\bar{L} = G\tau = (2 \times 10^{-8}\ \mathrm{m/s})(3600\ \mathrm{s}) = 7.2 \times 10^{-5}\ \mathrm{m} = 72\ \mu\mathrm{m}$$

A run twice as long (or grown at half the rate) doubles it — the direct lever a plant pulls when downstream filtration needs a coarser, faster-draining cake (§10.6 warm-up exercise below reproduces this exact number).

The mean crystal size

$$\bar{L}$$

falls right out as

$$G\tau$$

— a crystallizer run longer (or grown slower) makes bigger crystals, which is exactly the lever a plant pulls when downstream filtration needs a coarser, faster-draining cake. The solid production rate and the crystallization duty follow from the third moment and the (usually exothermic) heat of crystallization:

$$\dot{m}_{solid} = \rho_c\, k_v\, m_3, \qquad Q = -\dot{n}_{solid}\, \Delta H_{cryst}$$

$\dot{m}_{solid}$	Solid mass production rate leaving the crystallizer (kg/s).
ρ_c	Crystal (solid-phase) density — mass per unit volume of the pure crystal (kg/m³).
k_v	Volumetric shape factor — how efficiently the crystal shape fills its bounding cube (k_v = 1 for a perfect cube; real crystals pack less efficiently, so k_v < 1).

$\dot{n}_{solid}$	Molar solid production rate (mol/s) — the same physical flow as ṁ_solid, in moles rather than mass.
ΔH_{cryst}	Molar heat of crystallization (J/mol) — usually exothermic (negative), so Q comes out as heat that must be removed to hold the crystallizer at temperature.

$$k_v$$

is a shape factor (a cube has

$$k_v = 1$$

; real crystals are less efficient packers). MaximaLabs' **crystallizer** unit op solves exactly this moment model — set the saturation concentration, feed concentration, residence time, and heat of crystallization, and it returns the solid stream (with its mean size) and the mother liquor. Beyond the moment model, a **distributed population-balance** mode (a full size-grid solve with an optional agglomeration kernel) is also available for cases where the moment model's implicit single-mode distribution isn't enough — see the "further reading" note at the end of this chapter.

10.2 Solid-liquid separation: cake filtration

Crystals leave the crystallizer suspended in liquor — a **filter** (or centrifuge) splits that suspension into a wet solid cake and a clear filtrate. Two things matter: how much solid actually reports to the cake (the **recovery**) and how much liquid stays trapped in it (the **cake moisture**):

$$\text{cake} = R\,\dot{n}_s + \dot{n}_{liq,ret}, \qquad m = \dot{n}_{liq,ret}/\text{cake}$$

R	Specified solids recovery — the fraction of the feed's solid that reports to the cake (0–1).
m	Specified cake moisture fraction — the fraction of the cake's total mass that is retained liquid.
$\dot{n}_s$	Molar (or mass) flow of solid arriving at the filter in the feed suspension.
$\dot{n}_{liq,ret}$	Liquid flow retained (trapped) in the cake — everything else in the feed liquid reports to the filtrate.

where

$$R$$

is the specified solids recovery and

$$m$$

the specified cake moisture fraction — the rest of the liquid, plus any unrecovered solid, reports to the filtrate. Sizing the actual filter area for a given cake filtration time

$$t_c$$

and pressure drop

$$\Delta P$$

uses the classic **Ruth** cake-filtration equation, built on the same specific cake resistance

$$\alpha$$

and slurry concentration

$$c$$

a lab filter-leaf test measures:

$$A = Q_f \sqrt{\frac{\mu \, \alpha \, c \, t_c}{2 \, \Delta P}} \qquad \text{(Ruth cake filtration)}$$

A	Required filter area (m²) — the unknown this equation is solved for.
Q_f	Volumetric feed (slurry) flow rate to the filter (m³/s).
μ	Filtrate (liquid) viscosity (Pa·s).
α	Specific cake resistance — how much a unit mass of deposited cake resists flow (m/kg); a lab filter-leaf test measures this per system, it isn't a universal constant.
c	Slurry solids concentration — mass of solid per unit volume of filtrate (kg/m³).
t_c	Cake filtration time — how long the filter runs per cycle/batch (s).

ΔP	Pressure drop driving flow through the cake and filter medium (Pa).

A centrifuge separates the same suspension by replacing gravity with centrifugal acceleration —

$$\Sigma$$

theory converts a centrifuge's geometry and rotation speed into an equivalent gravity-settler area, letting you compare machines on the same basis:

$$v_g = \frac{G\,g\,(\rho_s - \rho_l)\,d^2}{18\,\mu}, \qquad \Sigma = \frac{Q_f}{2\,v_g}$$

v_g	Gravity-settling (terminal) velocity of a particle of diameter d under Stokes' law (m/s).
G	g-force multiple — the centrifuge's rotational acceleration divided by gravitational acceleration (dimensionless), from its bowl radius and rotation speed.
g	Gravitational acceleration, 9.81 m/s².
$\rho_s,\ \rho_l$	Solid particle density and liquid (continuous-phase) density (kg/m³) — their difference is the buoyancy-corrected driving force.
d	Particle diameter (m).
Σ	Equivalent settling area — the gravity-clarifier area that would give the same separation performance as the centrifuge, letting different machines be compared on one basis (m²).

10.3 Drying: the last of the liquid

The filter cake still carries the specified moisture — a **dryer** evaporates the rest down to a target outlet moisture

$$m_{out}$$

. The material balance on the liquid picks off exactly the evaporated amount

$$E$$

, and the duty is that evaporation rate times the latent heat of vaporization at the drying temperature:

$$E = L_{in} - S\,\frac{m_{out}}{1 - m_{out}}, \qquad Q = E\,\big(h_v(T) - h_l(T)\big)$$

E	Evaporated liquid flow — the amount removed from the cake to hit the target outlet moisture (mol/s).
L_{in}	Incoming liquid flow carried in the feed cake.
S	Dry-solid flow — constant through the dryer (solids don't evaporate).
m_{out}	Target outlet moisture fraction (liquid mass / total mass) — the spec the dryer solves E for.
Q	Drying duty — the heat input required (W).
$h_v(T),\ h_l(T)$	Vapor- and liquid-phase molar enthalpy of the liquid component at the drying temperature T, from the active thermo package — their difference is the latent heat of vaporization.

$$S$$

is the (constant, non-evaporating) dry-solid flow and

$$L_{in}$$

the incoming liquid — so the dryer solves for how much water has to leave to hit the target moisture, and prices that in real latent-heat duty from the active thermo package, not a rule-of-thumb energy number.

10.4 Sizing and classification: mills, screens, cyclones

Not every solids problem is about removing liquid — sometimes the goal is changing particle size, or splitting a distribution by size. A **mill** (comminution) reduces particle size and its power draw follows **Bond's law**, empirically relating the specific energy to the 80%-passing sizes before (

$$F_{80}$$

) and after (

$$P_{80}$$

) grinding:

$$W = 10\,W_i\left(\frac{1}{\sqrt{P_{80}}} - \frac{1}{\sqrt{F_{80}}}\right) \qquad \text{(Bond work index)}$$

W	Specific grinding energy required (kWh/t).
W_i	Bond work index — an empirical, material-specific measure of grindability (kWh/t), measured by a standard lab grinding test; higher means harder to grind.
$F_{80},\ P_{80}$	80%-passing particle size of the feed and product — the size below which 80% of the material's mass lies, before and after grinding (μm), Bond's chosen way to characterize a whole size distribution with one number.

A **screen** splits a particle-size distribution at a cut size using a log-normal partition (sharper for a well-designed screen, fuzzier for a real one via

$$\sigma_g$$

), and a **cyclone** separates fine solids from a gas by centrifugal action, characterized by its cut diameter

$$d_{50}$$

(the Lapple correlation) and a partition efficiency that falls off around it:

$$F_{under} = \Phi\left(\frac{\ln(c/d_{50})}{\ln \sigma_g}\right) \qquad \text{(screen partition)}$$

$$d_{50} = \sqrt{\frac{9\mu W}{2\pi N_e v\,(\rho_p - \rho_g)}}, \qquad \eta = \frac{1}{1 + (d_{50}/d_{eff})^2} \qquad \text{(cyclone, Lapple)}$$

F_{under}	Partition fraction reporting to the undersize (screened-through) stream, for a particle of size c.
Φ	Standard normal cumulative distribution function — the log-normal partition curve's S-shape comes from evaluating it in log-size space.
c	Particle size being partitioned (the screen equation's own variable, not the slurry concentration used earlier in §10.2).
d_{50}	Cut diameter — the particle size at which the screen (or cyclone) splits 50/50 between the two outlet streams.

σ_g	Geometric standard deviation of the partition curve — how sharp (near 1) or fuzzy (larger) the cut is; a real screen is never a perfect knife-edge.
μ	Gas viscosity, for the cyclone (Pa·s).
W	Cyclone inlet width (m) — reuses the same symbol as Bond's specific energy above but in a different unit-op context, per the Lapple correlation's own notation.
N_e	Effective number of turns the gas makes inside the cyclone body (dimensionless, geometry-dependent).
v	Cyclone inlet gas velocity (m/s).
$\rho_p,\ \rho_g$	Particle and gas density (kg/m³).
η	Collection (partition) efficiency for a particle of the effective diameter d_{eff}, falling off around d_{50}.

All four — mill, screen, cyclone, plus filter/centrifuge above — carry a full solid-phase payload (particle size, moisture, composition) on the stream itself, so a mill feeding a screen feeding a cyclone is a genuine size-tracking train, not three independent black boxes.

10.5 Try it

REPRODUCE IT IN YOUR BROWSER

1. Open the solids train example and **Run** it: FEED → **CR** (MSMPR crystallizer) → **FIL** (cake filter) → **DRY** (dryer).
2. Click the **CR** node and open the **Theory** tab — the exact nucleation/growth/moment equations of §10.1, with the solved mean crystal size and duty for this stream.
3. Raise the crystallizer's **residence time** and re-run — the mean crystal size

$$\bar{L} = G\tau$$

   grows, exactly as §10.1 predicts. Then check the **FIL** node: a coarser crystal typically filters faster and drier in a real plant, even though this screening-level filter model takes recovery/moisture as direct specs rather than deriving them from crystal size.
4. Tighten the dryer's **outlet moisture** spec and re-run — watch the dryer duty in the stream table rise, since more water now has to evaporate per §10.3.

The moment-based crystallizer above is a screening model — MaximaLabs also has a **distributed population-balance** mode with an optional agglomeration kernel

for cases where the crystal-size distribution itself (not just its mean and count) is the answer you need, e.g. sizing a filter to a real, possibly bimodal, distribution rather than an assumed mean.

10.6 Exercises

PRACTICE

Work each problem yourself first, then reveal the solution to check it. Where a problem says so, reproduce it live in MaximaLabs — the solver is the answer key.

1 **WARM-UP**

An MSMPR crystallizer has growth rate

$$G = 2 \times 10^{-8}$$

m/s and a residence time

$$\tau = 3600$$

s. Estimate the mean crystal size

$$\bar{L}$$

.

Show solution

2 **CORE**

Two crystallizers hit the *same* mean crystal size

$$\bar{L} = G\tau$$

— one by running twice as long (

$$\tau$$

) at half the growth rate

$$G$$

, the other by running half as long at double the growth rate. Are their crystal-count populations

$$m_0 = B\tau$$

also the same? What does that imply about their filtration behavior even at equal mean size?

Show solution

3 CHALLENGE

In the solids train, the dryer's outlet-moisture spec is tightened from 10% to 1%. Using §10.3's equation, explain qualitatively why the marginal evaporation duty needed per percentage point gets *larger* as the target moisture approaches zero, not constant.

Show solution

CHAPTER 11

Case Studies

Ten chapters have each covered one piece — a flash, a column, a reactor, a heat exchanger. Real processes chain many of these together to solve a problem no single unit can. This chapter walks four such flowsheets end to end, each a genuine running MaximaLabs example, not a narrated screenshot from someone else's simulator.

11.1 Breaking an azeotrope: extractive distillation

Chapter 5 (§5.6) showed the wall a plain column hits: ethanol-water can't be pushed past its ~89 mol% azeotrope by ordinary distillation, no matter how many stages you add. Here's the fix in practice — a reference case from Luyben (*Ind. Eng. Chem. Res.* 2006, 45, 4625). An 85 mol% ethanol feed enters an **extractive column** alongside a heavy, high-boiling **entrainer** (ethylene glycol) fed a few stages below the top:

- **EXTCOL** — 16 stages, two feeds (the ethanol-water mixture on stage 12, the glycol entrainer on stage 3). The glycol raises water's relative volatility enough to pull the overhead ethanol past the azeotrope in a single pass — nothing about the MESH equations from Chapter 5 changes, only that a third, non-volatile component reshapes the vapor-liquid equilibrium enough to open a path the binary system didn't have.
- **RECOVCOL** — an 8-stage column that strips the water back overhead from the EXTCOL bottoms (glycol + water), regenerating clean glycol as bottoms for recycle.

This is why entrainer selection matters as much as stage count: the entrainer has to be heavy enough to stay in the bottoms of the first column (so it doesn't contaminate the ethanol product) yet volatile enough relative to water to be strippable in the second. Run it and compare the EXTCOL distillate composition against the plain-column azeotrope ceiling from Chapter 5 — this is a genuinely different equilibrium surface, computed by the same NRTL package (Chapter 2), not a workaround bolted onto the solver.

11.2 Reaction, separation, and a second reaction: ethylene oxide to glycol

A single flowsheet chaining reaction, phase separation, a second reaction, and distillation — the shape of a real petrochemical train. Ethylene and oxygen feed two competing reactors over a silver catalyst: the desired **epoxidation** (ethylene → ethylene oxide) and an undesired **total combustion** side reaction (ethylene → CO_2 + water), modeled here as two sequential conversion reactors sharing the same ethylene — a deliberately low 8% per-pass conversion on the first reactor, because pushing per-pass conversion higher is exactly what favors the combustion side reaction in a real EO plant:

$$C_2H_4 + \tfrac{1}{2}O_2 \longrightarrow C_2H_4O \qquad (\text{epoxidation, desired})$$

$$C_2H_4 + 3\,O_2 \longrightarrow 2\,CO_2 + 2\,H_2O \qquad (\text{combustion, undesired})$$

A cooled flash (**CONDENSE**) then does the separation job Chapter 3 covers: condensing ethylene oxide and reaction water out of the unreacted ethylene/oxygen/CO_2, which vents for purge (real plants would recycle the unreacted ethylene at high ratio — left out here as a stated simplification, not a hidden one). The condensed EO stream mixes with fresh water and hydrates to monoethylene glycol in a second reactor:

$$C_2H_4O + H_2O \longrightarrow C_2H_6O_2 \qquad (\text{hydration, } 99\% \text{ conversion here})$$

A final distillation column (**COL**) then separates the water byproduct from the glycol product — the exact same rigorous MESH solve as the ethanol-water column, just purifying a different pair. Four unit types from four different chapters (reactor, flash, reactor, distillation), one flowsheet, one converged solve.

The two competing reactors together decide what actually matters commercially: how much of the fed ethylene ends up as EO rather than burned to CO_2. With

$$X_1$$

the epoxidation reactor's conversion (on the fresh ethylene feed) and

$$X_2$$

the combustion reactor's conversion (on what's left after epoxidation), the overall **selectivity** to EO is:

$$S = \frac{X_1}{X_1 + (1 - X_1)\,X_2}$$

X_1	Epoxidation-reactor conversion — fraction of fresh ethylene converted to EO in the first reactor.
X_2	Combustion-reactor conversion — fraction of the remaining ethylene (after epoxidation) burned to CO_2 + water in the second reactor.
S	Overall selectivity to EO — moles of ethylene that became EO ÷ total moles of ethylene reacted (both pathways).

WORKED EXAMPLE — THIS FLOWSHEET'S SELECTIVITY

With the flowsheet's

$$X_1 = 0.08$$

and

$$X_2 \approx 0.0217$$

:

$$S = \frac{0.08}{0.08 + 0.92 \times 0.0217} \approx 0.80$$

About 80% of the ethylene reacted ends up as EO, 20% burned — the deliberately low 8% per-pass conversion on EPOXRX is what keeps that ratio favorable (§11.6's exercise derives this same result from first principles).

11.3 A continuous polymerization reactor

The reference book's resin-reactor case study asks a different kind of question than the others: not just conversion and duty, but the **molecular-weight distribution** of the product itself — the number a specialty-polymer plant actually sells against. MaximaLabs' **polymerization** unit op is a continuous free-radical CSTR solved by an Arrhenius rate law closed with the **method of moments**, reporting the number- and weight-average molecular weights and the polydispersity index directly:

$$PDI = \frac{M_w}{M_n} \qquad \text{(a measure of chain-length spread, not just average size)}$$

M_n	Number-average molecular weight — the simple mean chain length (total polymer mass ÷ total number of chains).
M_w	Weight-average molecular weight — the mean weighted by each chain's own mass, so longer chains count more; always ≥ M_n.
PDI	Polydispersity index M_w/M_n — 1.0 is a perfectly uniform chain length; larger values mean a broader spread of chain lengths.

Two reactors can hit the identical monomer conversion and still ship a very different product if their

$$PDI$$

differs — a broader distribution means more short chains (weaker mechanical properties) and more very-long chains (harder to process) than a narrow one at the same average. **Honest scope**: the example flowsheet below runs the polymerization kinetics on a generic monomer/solvent pair rather than nylon-6,6's actual step-growth condensation chemistry (the book's specific case is a different polymerization mechanism entirely — batch step-growth, not continuous free-radical) — what transfers is the*reactor-design* question the book is really asking: given a CSTR, what molecular- weight distribution comes out, and how does it move with residence time and initiator concentration.

11.4 Economic evaluation

A converged flowsheet answers "does it work"; a cost estimate answers "is it worth building." MaximaLabs' capital-cost estimator (Turton-style correlations — purchased equipment cost from a size attribute per unit, escalated by material, location, and CEPCI, then multiplied by a bare-module installation factor) runs the same role as Aspen Icarus in the book's own case study, without a separate application: purchased cost by unit type comes from a power-law correlation on that unit's real solved size —

$$C_{purchased} = a\,(\text{size})^{\,b} \qquad \text{(e.g. compressor: } 7900\,(\text{kW})^{0.62}\text{)}$$

$$C_{installed} = C_{purchased} \cdot F_{BM} \cdot \frac{\text{CEPCI}_{now}}{\text{CEPCI}_{base}} \cdot (\text{location} \times \text{material} \times \text{year})$$

$C_{purchased}$	Base purchased-equipment cost, from the correlation alone (no installation labor/piping/instrumentation).
$a,\ b$	Correlation constants fit per unit type (e.g. a=7900, b=0.62 for a compressor) — b<1 is the "six-tenths rule": cost rises slower than size (see the worked example below).
size	The unit's own solved sizing attribute driving cost — duty for a compressor/heater, area for a heat exchanger, diameter×height for a column.
F_{BM}	Bare-module factor — installed cost ÷ purchased cost (e.g. 4.0 for a column/reactor, 2.2 for a heater), covering piping, instrumentation, and installation labor.
$\text{CEPCI}_{now},\ \text{CEPCI}_{base}$	Chemical Engineering Plant Cost Index at today's date vs. the correlation's original publication date — escalates a decades-old correlation to current cost levels.
$\text{location} \times \text{material} \times \text{year}$	Further multipliers for build location (e.g. Gulf Coast vs. Asia), construction material (carbon steel vs. stainless), and project year — all applied on top of the CEPCI escalation.

with the bare-module factor

$$F_{BM}$$

(installed cost ÷ purchased cost, e.g. 4.0 for a column or reactor, 2.2 for a heater) carrying the piping/instrumentation/labor allowance a purchased-equipment quote alone doesn't. Run it against any of this chapter's flowsheets — the compressor from Chapter 9, the reactors and column from §11.2 — and every installed-cost number traces back to that unit's own solved duty or flow, not a lookup table keyed on "distillation column."

WORKED EXAMPLE — WHY ONE BIG COMPRESSOR BEATS TWO SMALL ONES

Take the compressor correlation

$$C = 7900\,(\text{kW})^{0.62}$$

and split one unit's duty into two equal units in series, each at half the total kW. Purchased cost scales as

$$(\text{size})^{0.62}$$

, so two half-size units together cost

$$2 \times 2^{-0.62} \approx 1.30$$

times the single full-size unit — **~30% more** in total purchased cost for the same total duty. The sub-linear exponent

$$b < 1$$

is exactly why plants default to one train and only split equipment when something else forces it (§11.6's exercise works this out in full).

Honest scope (stated directly in the tool, not buried): these are AACE screening-level (order-of-magnitude) correlations from a published textbook, not a live vendor quote — the same caveat the book itself makes about Icarus's own accuracy class.

11.5 Try it

REPRODUCE THESE IN YOUR BROWSER

1. Open extractive distillation and **Run** — check the EXTCOL distillate composition against the ~89 mol% ceiling from Chapter 5's plain ethanol-water column.
2. Open ethylene oxide to glycol and **Run** — click **EPOXRX** vs. **COMBRX** to see the two competing reactors' duties and conversions side by side.
3. Open free-radical polymerization and **Run** — the **POLY** node's Theory tab reports Mn, Mw, and PDI. Raise the initiator concentration and re-run to see the distribution shift.
4. On any solved flowsheet, open **Analysis & reports ▸ Cost estimator** to get an installed-capital breakdown by unit, with the location/material/year adjustment factors from §11.4.

11.6 Exercises

PRACTICE

Work each problem yourself first, then reveal the solution to check it. Where a problem says so, reproduce it live in MaximaLabs — the solver is the answer key.

1 WARM-UP

In §11.2's ethylene-oxide reactors, the desired epoxidation and the undesired combustion reaction both consume ethylene. Write the overall **selectivity** expression (moles of ethylene to EO ÷ total moles of ethylene reacted) in terms of the two reactors' individual conversions

$$X_1$$

(epoxidation, on the fresh feed) and

$$X_2$$

(combustion, on the epoxidation reactor's outlet).

Show solution

2 CORE

Explain, using §11.1's physics, why the entrainer (ethylene glycol) has to be fed *above* the main feed stage in the extractive column rather than at the same stage or below it.

Show solution

3 CHALLENGE

§11.4's compressor cost correlation is

$$C = 7900\,(\mathrm{kW})^{0.62}$$

. A plant is considering replacing one large compressor with two smaller ones in series, each doing half the total duty. Using the correlation's exponent alone (ignoring bare- module and installation factors), is the two-compressor option cheaper or more expensive in purchased-equipment cost, and by roughly what factor? What does this imply about why plants don't always split equipment into many small parallel/series units?

Show solution