basicstyle=, identifierstyle=, commentstyle=, keywordstyle=, ndkeywordstyle=, stringstyle=, frame=tb, breaklines=true, columns=[l]fullflexible, numbers=left, xrightmargin=0zw, xleftmargin=3zw, numberstyle=, stepnumber=1, numbersep=1zw, lineskip=-0.5ex

令和元年度プロセス設計

トルエンの空気酸化による安息香酸の製造

1講座 移動現象論分野

荊尾太雅 宮本奏汰

Keywords

空気酸化	Air Oxidation
気液反応	Gas-liquid Reaction
晶析	Crystallization
多変数同時全体最適化	Multi-variable Simultaneous Optimization
ヒートインテグレーション	Heat Integration

目次

第 1 章

緒言

安息香酸は，主としてフェノールの原料となる他，その誘導体やそのエステルが食品や化粧品などの添加物として広く利用されている．2014 年には世界全体で 48 万トンが製造されており，新興国での需要の増加から，2024 年には生産量が 64 万トンとなることが見込まれている [1]．そこで，安価なトルエンを原料として用いて安息香酸を製造するプロセスを検討することにした．

第 2 章

プロセスの概要

2.1　プロセスの概要

本設計で対象とするのは，トルエンを空気酸化することにより安息香酸を製造するプロセスである．プロセス全体の概略図を図 2.1 に示す．原料となるトルエンは反応部で空気酸化し, 安息香酸, ベンズアルデヒドおよびベンジルアルコールに転化する．反応物は分離部 1,2 に送り, ベンズアルデヒドおよびベンジルアルコールを分離回収し, リサイクルする. また, 反応部で蒸発したトルエンの内, 凝縮回収しきれなかった分は燃焼部に送られる．

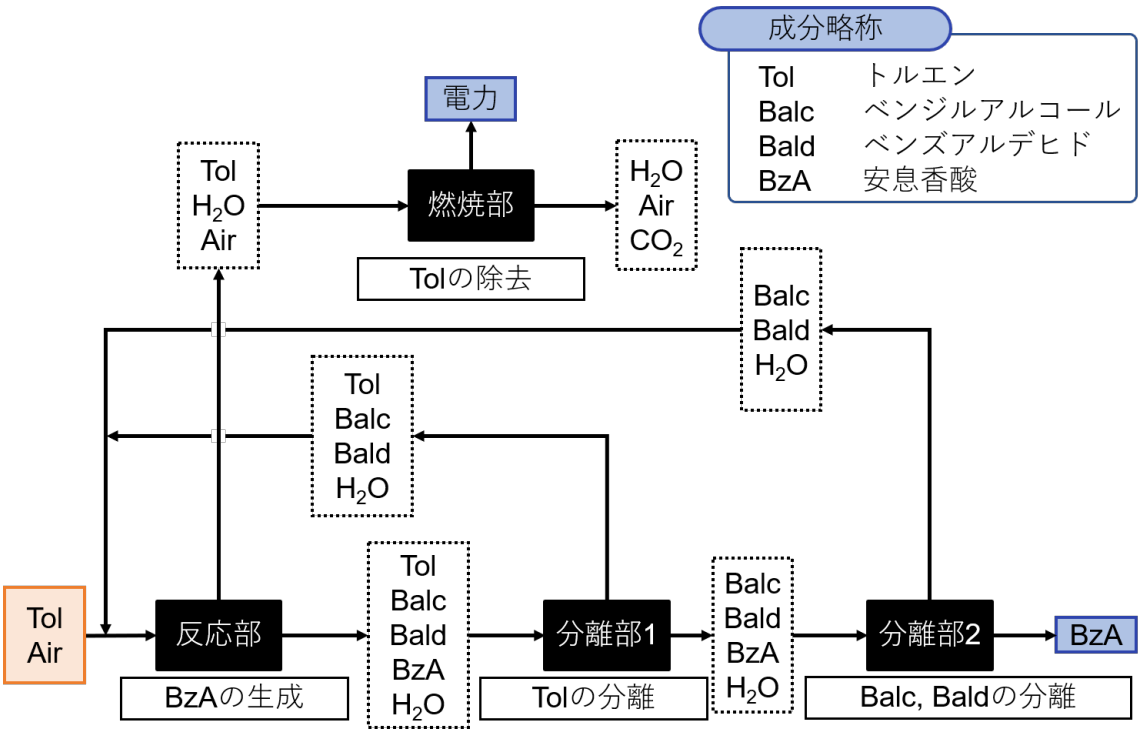

図 2.1　プロセス全体の概略図

2.2　設計条件

プロセスを設計するにあたり，以下の条件を設定した．

1. 生産要求は，99.0 wt% 以上の安息香酸を年 2 万トンとする．

2. 工場の稼働時間は，1 日 24 時間，年 300 日とする．

3. 原料として，純度 100 % のトルエンおよび，組成を窒素 79 mol% 酸素 21 mol% とする空気を用いる．ただし，両原料は 25 °C，1 bar で供給されるものとする．

4. 減価償却期間は 8 年とする．

5. 圧力損失，熱損失，制御系については考慮しない．

6. HYSIS を用いての物性推算は UNIQUAC 式によって行う．

第 3 章

反応部

トルエンを空気酸化し，安息香酸に転化する工程である．中間生成物としてベンジルアルコールとベンズアルデヒドができ，副生成物として水が生じる. 反応は触媒の溶解した液相中で生じる. 反応器は安息香酸，トルエンを液相中に保つための適切な温度, 圧力で運転する必要がある．反応部の概略図を図 3.1 に示す．反応器にフィードされる液は，原料およびリサイクルによって回収された未反応トルエン，ベンジルアルコール，ベンズアルデヒドと水であり、反応器入り口において 7 bar，170 °C に加熱および加圧され，反応器に供給される．また，反応器底部からは空気が 1 bar，25 °C で供給されており，攪拌されながら反応器上方に向かう．反応器内は外部熱煤によって加熱され，7 bar，170 °C に保たれており，攪拌によって液中に溶けた酸素と未反応物質は触媒酸化反応を起こす．また，気液間物質移動により，液中のトルエンおよび水が盛んに蒸発するが，蒸発したトルエンおよび水を含む空気は反応器からコンデンサーに流入し凝縮される．凝縮したトルエンおよび水はデカンターへ送られ，密度差分離によりトルエンのみを反応器に還流し，水はパージする．また，凝縮しなかったトルエンは，環境安全上の理由からそのまま排出することはできないので，濃度を減少させるために燃焼部へと送られる．反応器からの流出液は次の工程である分離部 1 の蒸留塔へと送られる．

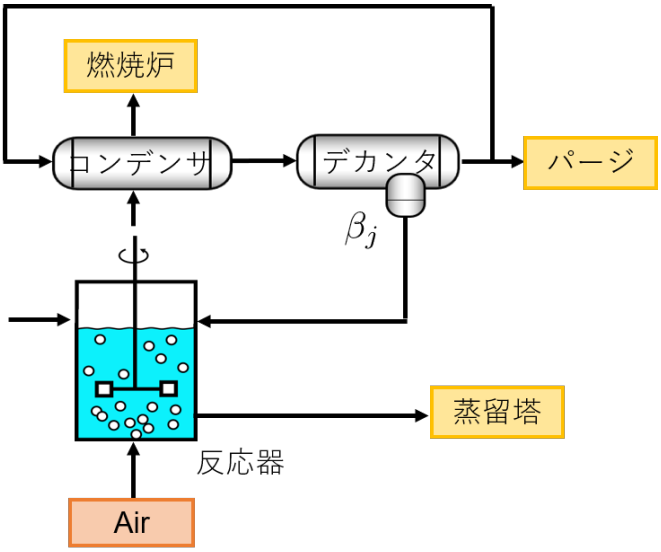

図 3.1　反応部概略図

3.1　反応機構

今回用いた反応を以下に示す．それぞれの反応速度式および，298 K における反応エンタルピーも示している．トルエンは経路によらず最終的に安息香酸へと転化する．

$$
\begin{array}{lcrll}
\mathrm{Tol} & + & \mathrm{O_2} & \longrightarrow \mathrm{Bald} + \mathrm{H_2O} & r_1 = k_1 C_{\mathrm{Tol}} \quad \Delta_{\mathrm{r}} H_1 = -385.23\,\mathrm{kJ\,mol^{-1}} \\
\mathrm{Tol} & + & 1/2\ \mathrm{O_2} & \longrightarrow \mathrm{Balc} & r_2 = k_2 C_{\mathrm{Tol}} \quad \Delta_{\mathrm{r}} H_2 = -167.30\,\mathrm{kJ\,mol^{-1}} \\
\mathrm{Balc} & + & 1/2\ \mathrm{O_2} & \longrightarrow \mathrm{Bald} + \mathrm{H_2O} & r_4 = k_3 C_{\mathrm{Balc}} \quad \Delta_{\mathrm{r}} H_3 = -317.30\,\mathrm{kJ\,mol^{-1}} \\
\mathrm{Bald} & + & 1/2\ \mathrm{O_2} & \longrightarrow \mathrm{BzA} & r_3 = k_4 C_{\mathrm{Bald}} \quad \Delta_{\mathrm{r}} H_4 = -298.20\,\mathrm{kJ\,mol^{-1}}
\end{array}
$$

ただし，トルエンを Tol，ベンジルアルコールを Balc，ベンズアルデヒドを Bald，安息香酸を BzA と表記している．これらの反応は全て酸化反応であり，触媒にはモリブデン酸マンガン $\mathrm{MnMoO_4}$ を用いることとした．これらの反応の反応速度定数はアレニウス式に従い，パラメータを表 3.1 に示す [2]．

$$
k_j = A_j \exp(-\frac{E_j}{RT}) \tag{3.1}
$$

表 3.1　各反応式の反応速度定数

$k_j\ [\mathrm{s^{-1}}]$	k_1	k_2	k_3	k_4
$\ln A_j\ [-]$	17.93	20.63	15.40	19.70
$E_j\ [\mathrm{kJ\,mol^{-1}}]$	69.53	81.39	56.99	71.44

また，酸化反応が激しい場合にはトルエンは二酸化炭素まで酸化される. 反応器における反応率が高い場合には考慮する必要があるが，今回の設計結果では単通反応率が 0.265 と低いため，この反応は十分無視できると考えられる．

3.2　反応器選定

流通式の気液反応器の種類には，拡散速度に対して不利な順に，気泡塔，気液攪拌槽，充填塔などがある．反応器を選定するにあたり，最大反応速度と最大拡散速度の比を表す八田数を概算し，八田数が 0.1 より小さいなら気泡塔，5 より大きければ充填塔，中間域ならば気液攪拌槽を選択することとした [3]．概算により八田数が 0.3 となったので今回は気液攪拌槽型反応器を選択した．実際のプロセスでは気泡塔も選択される．

反応器内攪拌には 6 枚羽根タービン翼を用い，空気流入による冷却作用が大きいため，ジャケットを取り付け，ジャケット内部に熱媒を流している．また，空気を反応器内に送るスパージャー直径は反応器直径の 1/3 とした．反応装置の L/D 比を 2 とした．また，反応器体積のうち半分は液相が占めるとした．

八田数の定義式

$$
\gamma = \frac{(\text{最大反応速度})}{(\text{最大拡散速度})} = \frac{(C_{\mathrm{B}} k D_{\mathrm{B}})^{1/2}}{k_{\mathrm{L}}} \tag{3.2}
$$

3.3　設計方程式

両相の滞留時間の大きさが十分異なると考えられることから，液相は完全混合状態として，気相は鉛直方向に向かう押し出し流れと仮定できると判断した．設計時に用いた仮定は以下のようである．

- 気相は水平方向に一様な濃度分布を持つ.
- 気相側境膜抵抗は無視できる
- 液相は完全混合状態である.
- 窒素，酸素はヘンリー則に従い，その他の物質はラウール則に従う.

以上の仮定および，蒸発油分を還流する機構を含めて，設計方程式 (3.3)，(3.4)，(3.5) を立式した.

$$0 = F_{\mathrm{liq},j}^{\mathrm{in}} - F_{\mathrm{liq},j}^{\mathrm{out}} - (1-\beta_j)k_{\mathrm{L}}a\int_0^{V_{\mathrm{tot}}}(C_j - C_j^{\mathrm{sat}})\mathrm{d}V + r_j V_{\mathrm{L}} \tag{3.3}$$

$$\frac{\mathrm{d}F_{\mathrm{gas},j}}{\mathrm{d}V} = k_{\mathrm{L}}a(C_j - C_j^{\mathrm{sat}}) \tag{3.4}$$

$$\sum_j F_{j,\mathrm{in}}H_{j,\mathrm{in}} - \sum_j F_{j,\mathrm{out}}H_{j,\mathrm{out}} = UA(T_{\mathrm{s}} - T) \tag{3.5}$$

解析方法としては，まず油分を全還流として近似的に各油分の液相中濃度を決定した．そして，酸素，窒素について液中濃度を仮定した後に，気相の物質収支式を Runge-Kutta 法によって計算し，各蒸発量が収支式と一致する濃度を求めた．また，物質移動容量係数が十分大きいと判断し，最適化計算を行う場合には，気相は迅速に平衡状態へ達すると仮定した.

3.4 物質移動容量係数の推算

物質移動容量係数は，反応装置形状，反応器内部流体の様々な物性，流れの状態に依存する．撹拌槽型反応器に対し，物質移動容量係数の推算に用いた各相関式を記す.
液相側物質移動係数の相関式 [3]

$$\text{小気泡の場合：}\quad k_{\mathrm{L}} = 0.31Sc_{\mathrm{L}}^{-2/3}(g\Delta\rho\mu_{\mathrm{L}}/\rho_{\mathrm{L}}^2)^{1/3} \tag{3.6}$$

$$\text{大気泡の場合：}\quad k_{\mathrm{L}} = 0.42Sc_{\mathrm{L}}^{-1/2}(g\Delta\rho\mu_{\mathrm{L}}/\rho_{\mathrm{L}}^2)^{1/3} \tag{3.7}$$

比表面積の相関式 [3]

$$a = 1.44\left(\frac{P_{\mathrm{V}}^{0.4}\rho_{\mathrm{L}}^{0.2}}{\sigma^{0.6}}\right)\left(\frac{u_{\mathrm{G}}}{u_{\mathrm{t}}}\right)^{0.5}\left(\frac{P_{\mathrm{T}}}{P_{\mathrm{G}}}\right)\left(\frac{\rho_{\mathrm{G}}}{\rho_{\mathrm{a}}}\right)^{0.16} \tag{3.8}$$

上記の 2 式を利用するために用いた相関式を以下に記す.
ガスホールドアップの相関式 [3]

$$\varepsilon_{\mathrm{G}} = \left(\frac{u_{\mathrm{G}}\varepsilon_{\mathrm{G}}}{u_{\mathrm{t}}}\right) + 0.000216 \times \left(\frac{P_{\mathrm{V}}^{0.4}\rho_{\mathrm{L}}^{0.2}}{\sigma^{0.6}}\right)\left(\frac{u_{\mathrm{G}}}{u_{\mathrm{t}}}\right)^{0.5}\left(\frac{P_{\mathrm{T}}}{P_{\mathrm{G}}}\right)\left(\frac{\rho_{\mathrm{G}}}{\rho_{\mathrm{a}}}\right)^{0.16} \tag{3.9}$$

気泡の体積平均径の相関式 [3]

$$d_{\mathrm{vs}} = 4.15\left(\frac{\sigma^{0.6}}{P_{\mathrm{V}}^{0.4}\rho_{\mathrm{L}}^{0.2}}\right)\left(\frac{P_{\mathrm{G}}}{P_{\mathrm{T}}}\right)\left(\frac{\rho_{\mathrm{a}}}{\rho_{\mathrm{G}}}\right)^{0.16}\varepsilon_{\mathrm{G}}^{0.5} + 0.0009 \tag{3.10}$$

気泡の終末速度の相関式 [3]

$$u_{\mathrm{t}} = \left(\frac{4\Delta\rho g d_{\mathrm{vs}}}{3C_{\mathrm{D}}\rho_{\mathrm{L}}}\right)^{0.5} \tag{3.11}$$

さらに，上記の相関式を利用するために用いた物性値の推算式，および変数の定義式などの諸式を以下に記す.

抗力係数の相関式 [4]

$$C_{\mathrm{D}} = \max\left[\frac{24}{Re}(1 + 0.15Re^{0.687}),\ \frac{8}{3}\frac{Eo}{Eo+4}\right] \tag{3.12}$$

拡散係数の推算

Wilke-chang の式 [5]

$$D_{12} = \frac{2.946 \times 10^{-11}(\beta M_{\mathrm{r},2})^{1/2} T}{\mu_2 V_{\mathrm{b},1}^{0.6}} \tag{3.13}$$

Einstin-Stokes の式 [6]

$$\frac{D\mu}{T} = \text{const.} \tag{3.14}$$

界面張力の推算

界面張力の温度依存性に関する相関式 [3]

$$\sigma \propto \{1 - (T/T_{\mathrm{c}})\}^n \tag{3.15}$$

気泡レイノルズ数の定義式 [4]

$$Re = \frac{\rho_{\mathrm{L}} u_{\mathrm{t}} d_{\mathrm{vs}}}{\mu_{\mathrm{L}}} \tag{3.16}$$

エトベス数の定義式 [4]

$$Eo = \frac{g\Delta\rho d_{\mathrm{vs}}^2}{\sigma} \tag{3.17}$$

また，空気の沸点における分子容を $29.9\,\mathrm{cm^3\,mol^{-1}}$ であり [3]，トルエンの界面張力を $\sigma = 0.81\,\mathrm{N\,m^{-2}}$ である [7]，会合度を 1 とみなせるとした [8].

以上の諸式を用いて物質移動容量係数を推算した．結果は反応器設計結果で示す．

3.5 反応器設計結果

最終結果における反応器の詳細を表 3.2 に示す．ただし，反応器内温度は，外部熱媒を用いることで 170 °C に保たれているとした．

3.6 反応部設計結果

最適化後の最終結果における反応部の物質収支を図 3.2 中に示す．最適化方法については最適化の章で述べる．

表 3.2　反応器設計結果

項目	値
反応器内圧力　[bar]	7.00
反応器内温度　[°C]	170
反応器体積　[m^3]	14.6
気液総体積　[m^3]	8.25
単通反応率　[–]	0.265
ガスホールドアップ　[–]	0.115
液平均滞留時間　[h]	0.478
物質移動容量係数　[s^{-1}]	0.470
液相物質移動係数　[$m\,s^{-1}$]	0.00172
比界面積　[m^{-1}]	273
反応器空間率　[–]	0.435
総攪拌動力　[kW]	8.25
気泡体積平均径　[mm]	2.53
エトベス数　[–]	1480
気泡レイノルズ数　[–]	1170

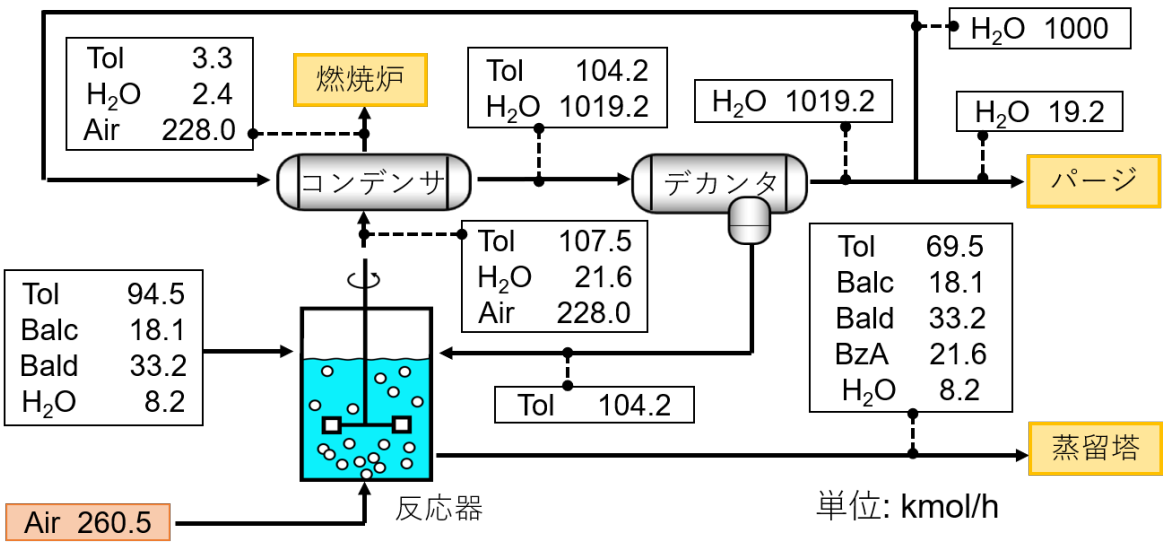

図 3.2　反応部設計結果

第 4 章

分離部 1

反応器からの流出液のうち，主に未反応トルエンを分離することを目的とする工程である．低沸点成分であるトルエンおよび水は蒸留によってほぼ全量が回収され，回収しきれなかったベンジルアルコールやベンズアルデヒド，安息香酸およびモリブデン酸マンガン触媒は分離工程 2 へと送られる．

4.1　蒸留塔設計

未反応トルエンのうち 99 % 以上を回収することを目的とした．設計条件として，蒸留塔段数を 10 段，蒸留塔供給段を 6 段，還流比を 1.0 とした．

蒸留塔圧力を変更し，全体の利益を最大とする点を探索した．反応器流出液圧力が 7 bar であるため，7 bar が最適な圧力であることが予想される．蒸留塔の圧力を変化させ，人件費を除くプロセス全体の利益を評価関数としてプロットした図 4.1 から，確かに 7 bar でプロセス全体の利益が最大化されることを確認した．

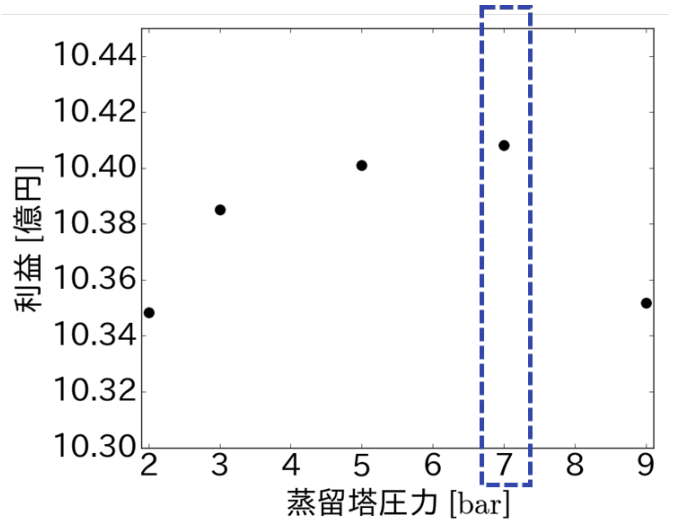

図 4.1　蒸留塔圧力最適化

設計結果を表 4.1 に記す．

表 4.1　蒸留塔設計結果

項目	値
塔径　[m]	1.0
塔高　[m]	6.1
塔内圧力　[bar]	7
コンデンサ内温度　[°C]	232
リボイラー内温度　[°C]	316

4.2　分離部 1 設計結果

最適化後の最終結果における分離部 1 の流量関係を図 4.2 中に示す.

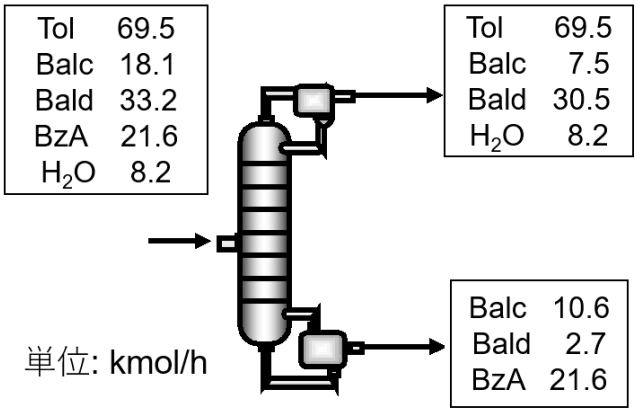

図 4.2　分離部 1 設計結果

第 5 章

分離部 2

他成分を分離し，安息香酸の製品結晶を得る工程である．蒸留塔の流出液に含まれるベンジルアルコール，ベンズアルデヒド，安息香酸，モリブデン酸マンガン触媒を抽出塔に送る．また，晶析後の安息香酸水溶液をリサイクルして同時に抽出塔に入れ，安息香酸とその他の成分を分離する．安息香酸飽和水溶液は晶析器に運ばれ，製品結晶を取り出す．その他の成分については回収し，反応器へと循環させる．

5.1　晶析器選定

結晶，溶液をともに連続的に流通させることができる連続式の晶析装置を選択した．溶解度の温度依存性が大きいことと，目的とする結晶生産量が大きいことから，連続式攪拌槽型晶析装置を選定した．結晶は針状であり，一般的な球形の結晶よりも装置内に詰まる恐れがある [3]．したがって，製品結晶のみが固体状で装置内を占めると想定して，空間率が 0.9 以上となるように十分な装置容積をとる必要があるため，晶析装置内液相部の 2 倍の装置容積を用意するものとして設計した．L/D 比は 2 とした．

溶解度の温度依存性の相関式

$$C^* = 2.03 \times 10^{-5} T^4 + 2.03^{-5} \times T^4 + 2.97 \times 10^{-4} T^3 + 4.70 \times 10^{-2} T^2 + 1.43T + 24.71 \tag{5.1}$$

ただし, 温度 T [°C]，溶解度 C^* [g kg-solvent^{-1}] である．

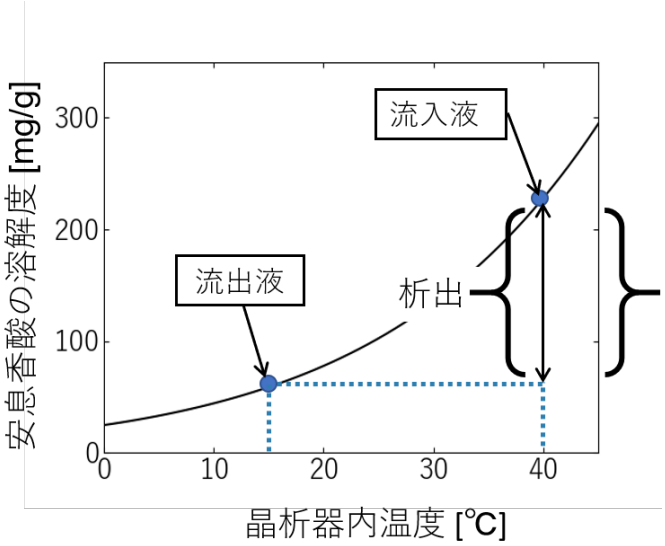

図 5.1　溶解度の温度依存性

5.2　設計方程式

以下の仮定を用いた．

- 結晶表面拡散は迅速に行われる．
- 晶析器内は完全混合状態である．
- 二次核発生の影響は無視する．

[9] により，以下の実験式および理論式を用いて設計を行った．

一次核発生速度

$$B^0 = k_\mathrm{b} M_\mathrm{T}^j \Delta C^b \tag{5.2}$$

結晶成長速度

$$G = k_\mathrm{g} \Delta C^g \tag{5.3}$$

結晶成長速度定数

$$k_\mathrm{g} = k_\mathrm{g0} \exp\left(-\frac{E_\mathrm{g}}{RT}\right) \tag{5.4}$$

個数収支式

$$n = n^0 \exp\left(-\frac{L}{G\tau}\right) \tag{5.5}$$

懸濁密度

$$M_\mathrm{T} = c_0 - c = 6 k_\mathrm{v} \rho_\mathrm{c} n^0 (G\tau)^4 \tag{5.6}$$

各パラメータは以下の通りである．

$$\begin{aligned}
&k_\mathrm{g0} = 1.06 \times 10^7 \,(\mu\mathrm{m})(\mathrm{g/g\text{-}solvent})^{\text{-}g} \\
&E^\mathrm{g} = 40.05\,\mathrm{kJ\,mol^{-1}} \\
&k_\mathrm{b} = 9.16 \times 10^{12} \,(\#/\mathrm{m^3 s})(\mathrm{g/mL})^{\text{-}j}(\mathrm{g/g\text{-}solvent})^{\text{-}b} \\
&g = 0.44 \\
&j = 1.78 \\
&b = 1.2 \\
&k_\mathrm{v} = 0.1 \\
&\rho_\mathrm{c} = 1.32\,\mathrm{g\,cm^{-3}}
\end{aligned}$$

5.3　晶析器設計結果

最適化の結果によって得られた晶析器の設計結果を表 5.1 に記す．最適化手法については最適化の章で説明する．

表 5.1　晶析器設計結果

項目	値
晶析器体積　[m^3]	7.16
晶析器液体積　[m^3]	3.58
晶析器フィード液温　[°C]	40.0
晶析器内温度　[°C]	13.3
晶析器内圧力　[bar]	1.00
滞留時間　[min]	9.71
単通結晶収率　[–]	0.690
結晶化可能量基準収率　[–]	0.902
結晶の体積平均径　[µm]	3.41

5.4　抽出塔設計

十分に塔内へ液を滞留させることによって，安息香酸を水中に飽和させることを目的とした．十分なデータを得られなかったため，2 時間の装置内滞留によって安息香酸が水中に飽和し，その他ベンジルアルコール，ベンズアルデヒドの溶解度については無視できると仮定した．

5.5　分離部 2 設計結果

分離部 2 のみの流量関係を簡易的に図 5.2 中に示す．蒸留塔から送られてきた油液は冷却され，抽出塔内に供給される．晶析器からリサイクルされる安息香酸水溶液を加熱し，溶解度を上昇させて抽出塔内に供給する．抽出塔内では安息香酸が水中に溶解し，ベンジルアルコール，ベンズアルデヒドおよび触媒は溶解せずに回収され，反応部へ送られる．安息香酸水溶液を晶析器に供給し，製品結晶を得ている．

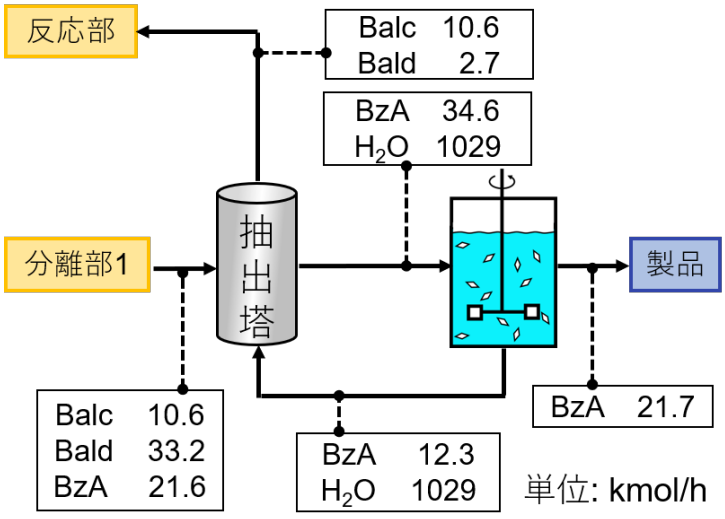

図 5.2　分離部 2 設計結果

第 6 章

燃焼部

トルエンは人体に有害な物質であり，吸引によって神経系などに深刻な被害を与えることがある．排出指針値としてトルエンを扱う作業環境において 50 ppm，その他の場所については 0.7 ppm と定められている [10]. この基準を満たすため，反応部のコンデンサーで凝縮しなかったトルエンを燃焼させることで，その濃度を減じる．燃焼炉内ではトルエンが完全燃焼し，二酸化炭素および水へ転化すると仮定し，量論的に燃焼に必要な量と等量の酸素を供給して燃焼炉へと送るものとした．燃焼熱は熱媒の生成に使用され，足りない分は燃料を投入して賄う.

第 7 章

最適化

7.1　方法

最適化を行うにあたり，最適化変数として以下の 3 つを選択した．

- 反応器体積
- 晶析器体積
- 晶析器温度

これらの変数を選択した理由を述べる．まず，反応器体積についてのトレードオフの関係を説明する．反応器体積が大きいとき，反応器の装置コストは大きくなる．一方で，反応器出口から排出される未反応トルエンの量が減るため，蒸留塔のリボイラーで必要な熱量が減少し，熱媒のコストは減少する．また，反応器体積が小さいときは，反応器の装置コストは小さくなるが，熱媒のコストは大きくなる．

次に，晶析器体積についてのトレードオフの関係を説明する．晶析器体積が大きいとき，晶析器の装置コストは大きくなる．しかし，析出する安息香酸の量が増えるため，リサイクルに回る安息香酸の量が少なくなり，それを保持するための純水の量が減少することで，純水の費用は抑えられる．晶析器の体積が小さいときは，逆の関係となる．

最後に，晶析器温度についてのトレードオフの関係を説明する．晶析器温度が低い時，必要な外部冷媒の温度が低くなるため，冷媒の単価は高くなる．しかし，晶析する安息香酸の量が大きくなるため，リサイクルに回る安息香酸の量が小さくなり，必要な純水の量が減るため，純水を冷やすために用いている冷媒の量は小さく抑えられる．晶析器温度が高いときは，冷媒の単価は安くなるが，冷媒の必要量が大きくなる．このように，最適化変数に選択した変数は，その大小によって，トレードオフの関係が生じる．そのため，これらの変数の値を変化させることで，利益が最大となる最適点を探索することが可能であると考えた．

次に，最適化の具体的な手法について述べる．プロセスを最適化するにあたり，3 つの最適化変数全てを同時に最適化することを考えた．運転可能な範囲を考慮して，反応器体積は，$13.5 \sim 16.0\,\mathrm{m}^3$ の範囲で $0.5\,\mathrm{m}^3$ 刻みで 6 点，晶析器体積は，$6 \sim 12\,\mathrm{m}^3$ の範囲で $0.5\,\mathrm{m}^3$ 刻みで 13 点，晶析器温度は，$5.0 \sim 20.0\,^\circ\mathrm{C}$ の範囲で $1\,^\circ\mathrm{C}$ 刻みで 16 点，計 1248 点のデータを取った．そして，それぞれのデータについてフィッティングを行った後に，それぞれの探索範囲を 100 等分し，データの個数を $100 \times 100 \times 100$ 点に増やした．それらのデータを用いて，横軸に晶析器温度，縦軸に晶析器体積を設定し，反応器体積の値を逐次的に変化させることで，100 枚のヒートマップを作成した．ヒートマップが表す値は，

$$\text{P.I.} = (\text{売上}) - (\text{原料コスト}) - (\text{装置コスト}) - (\text{用役コスト}) \tag{7.1}$$

で表される評価関数の値とした．また，各ヒートマップ上で評価関数の値が最大となる点をプロットした．さらに，評価関数の最大値がどのように変化するかを見るために，横軸に反応器体積，縦軸に評価関数の値を取

り，グラフを作成した．

7.2　最適化結果

評価関数の値が最大となった点でのヒートマップおよび，評価関数の最大値の変化を表したグラフを図 7.1 に示す．最適点での反応器体積は 14.69 m^3，晶析器体積は 7.16 m^3，晶析器温度は 13.3 °C となった．また，その時の評価関数の値は 10.35 億円となった．

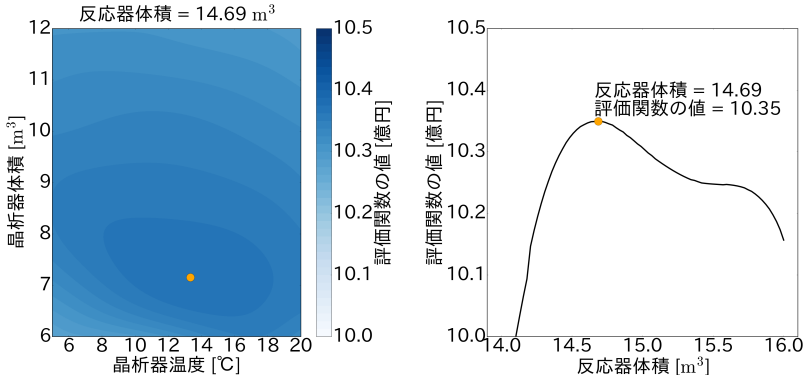

図 7.1　評価関数の値が最大となる点でのヒートマップと最適点の推移

このように，評価関数は大きなピークが存在する形になっていることが分かる．これは，最適化方法の節でも述べたが，反応器体積，晶析器体積，晶析器温度を変化させると，それぞれのトレードオフの関係によって，コストが変化する．そのため，このようにピークが生じたと考えられる．

第 8 章

物質収支・熱収支

全体のフロー図は図 8.1 のようになった.

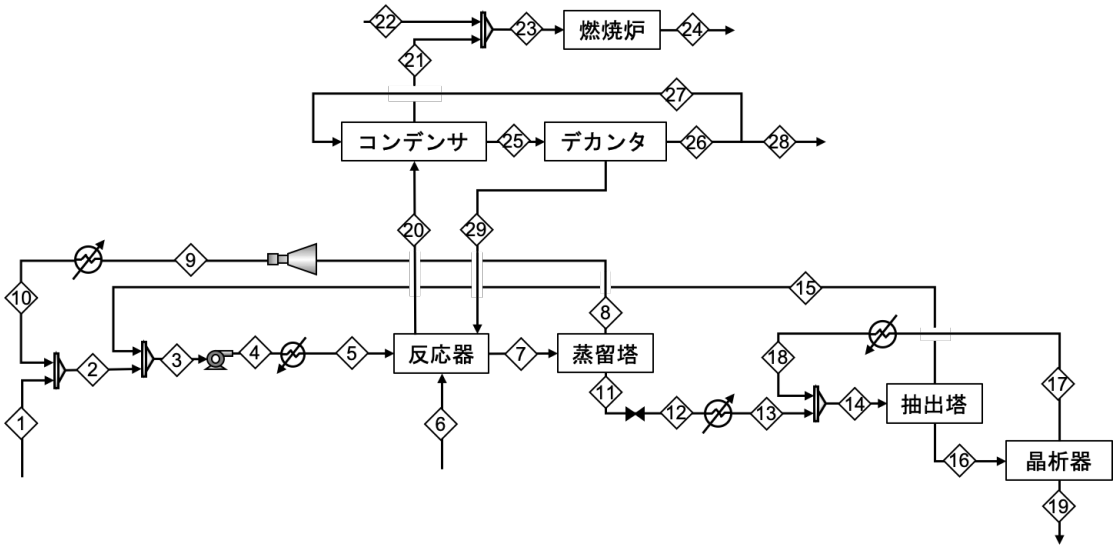

図 8.1　フロー図

また，フロー図に記載されているフローの組成および温度，圧力は表 8.1，8.2，8.3 のようになった.

表 8.1　流量関係

[kmol h^{-1}]	1	2	3	4	5	6	7	8	9	10
トルエン	25	94.5	94.5	94.5	94.5	0	69.5	69.5	69.5	69.5
ベンジルアルコール	0	7.5	18.1	18.1	18.1	0	18.1	7.5	7.5	7.5
ベンズアルデヒド	0	30.5	33.2	33.2	33.2	0	33.2	30.5	30.5	30.5
安息香酸	0	0	0	0	0	0	21.6	0	0	0
水	0	8.2	8.2	8.2	8.2	0	8.2	8.2	8.2	8.2
空気	0	0	0	0	0	260.5	0	0	0	0
二酸化炭素	0	0	0	0	0	0	0	0	0	0
合計	25	140.7	154	154	154	260.5	150.6	115.7	115.7	115.7
温度　[°C]	25	79.9	75.56	76.04	170	25	170	231.5	194.8	90.88
圧力　[bar]	1	1	1	7	7	1	7	7	1	1

表 8.2　流量関係

[kmol h^{-1}]	11	12	13	14	15	16	17	18	19	20
トルエン	0	0	0	0	0	0	0	0	0	107.5
ベンジルアルコール	10.6	10.6	10.6	10.6	10.6	0	0	0	0	0
ベンズアルデヒド	2.7	2.7	2.7	2.7	2.7	0	0	0	0	0
安息香酸	21.6	21.6	21.6	34.6	0	34.6	13	13	21.6	0
水	0	0	0	1029	0	1029	1029	1029	0	21.6
空気	0	0	0	0	0	0	0	0	0	228
二酸化炭素	0	0	0	0	0	0	0	0	0	0
合計	34.9	34.9	34.9	1076.9	13.3	1063.6	1042	1042	21.6	357.1
温度　[°C]	315.6	230.9	25	25	40	40	13.3	40	13.3	25
圧力　[bar]	7	1	1	1	1	1	1	1	1	1

表 8.3　流量関係

[kmol h^{-1}]	21	22	23	24	25	26	27	28
トルエン	3.4	0	3.4	0	104.1	0	0	0
ベンジルアルコール	0	0	0	0	0	0	0	0
ベンズアルデヒド	0	0	0	0	0	0	0	0
安息香酸	0	0	0	0	0	0	0	0
水	2.4	0	2.4	15.8	0	0	0	0
空気	228	39.4	267.4	237.8	1019.2	1019.2	1000	19.2
二酸化炭素	0	0	0	23.4	0	0	0	0
合計	233.8	39.4	273.2	277	1123.3	1019.2	1000	19.2
温度　[°C]	25	25	25	25	25	25	25	25
圧力　[bar]	1	1	1	1	1	1	1	1

フロー全体の与熱流体および，受熱流体は図 8.2 のようになった．図中の流体の熱量関係は表 8.4 および，8.5 のようになった．

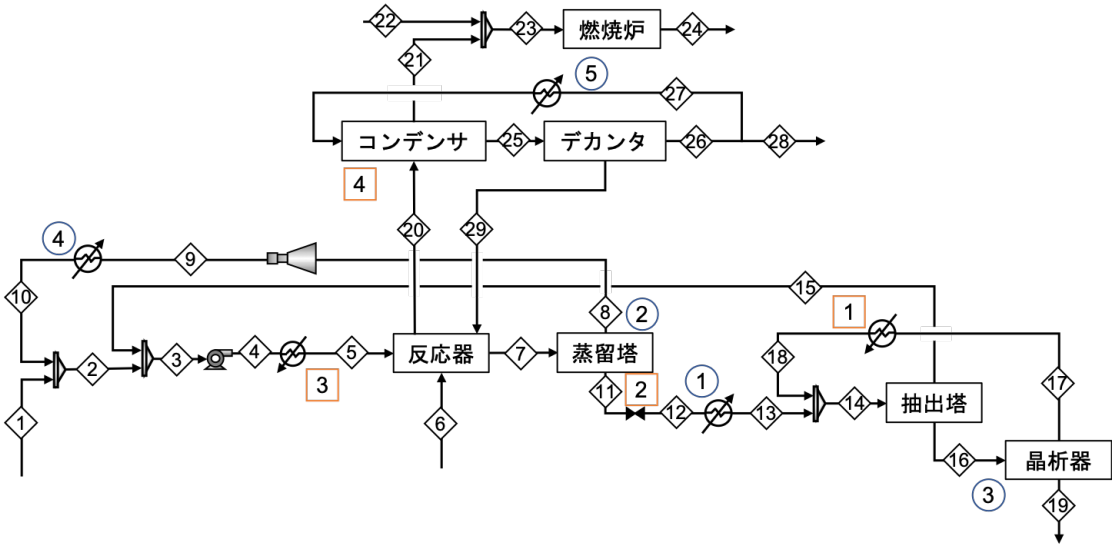

図 8.2　与熱流体と受熱流体

表 8.4　与熱流体

与熱流体	変化前温度 [°C]	変化後温度 [°C]	与熱量 [MJ h^{-1}]
① 蒸留塔留出液	194.8	90.9	6338.4
② 蒸留塔コンデンサー	248.3	231.5	5111.8
③ 晶析器流入流体	40.0	13.3	2252.7
④ 蒸留塔缶出液	230.9	25	2436.7
⑤ コンデンサ流出純水	40.0	25	1130.6

表 8.5　受熱流体

受熱流体	変化前温度 [°C]	変化後温度 [°C]	受熱量 [MJ h^{-1}]
[1] 晶析器リサイクル	13.3	40.0	2149.3
[2] 蒸留塔リボイラー	304.0	315.6	11735.8
[3] 反応器入り口流体	76.0	170.0	2922.4
[4] コンデンサ流入純水	25.0	40.0	1130.6

第 9 章

ヒートインテグレーション

流体同士の熱交換を行い，外部流体の利用量を削減することを目的としてヒートインテグレーションを行った．本プロセスにおいては熱交換器は多管型熱交換器として，向流で熱交換を行った．熱交換面積を求めるため，以下の式を用いた．

$$Q = UA(\Delta T)_{\mathrm{lm}} \tag{9.1}$$

ただし $(\Delta T)_{\mathrm{lm}}$ は温度差の対数平均であり，熱交換によって与熱流体の温度が T_{h1} から T_{h2} に変化し，受熱流体の温度が T_{c2} から T_{c1} に変化するとき，

$$(\Delta T)_{\mathrm{lm}} = \frac{(T_{\mathrm{h1}} - T_{\mathrm{c1}}) - (T_{\mathrm{h2}} - T_{\mathrm{c2}})}{\ln\{(T_{\mathrm{h1}} - T_{\mathrm{c1}})/(T_{\mathrm{h2}} - T_{\mathrm{c2}})\}} \tag{9.2}$$

と表される．総括熱伝達係数として，両熱交換流体の相状態にのみ依存するとして表 9.1 の値を用いた．

表 9.1　総括熱伝達係数

流体 1	流体 2	総括伝熱係数 [$\mathrm{W\,m^{-1}\,s^{-1}}$]
ガス	ガス	150
ガス	液	200
ガス	ガス (凝縮)	500
ガス	液 (蒸発)	500
液	液	300
液	ガス (凝縮)	1000
液	液 (蒸発)	1000
ガス (凝縮)	液 (蒸発)	1500

用いた外部熱媒の温度と熱量を表 9.2 に示す．

表 9.2　用いた外部熱媒

熱媒	温度 [°C]	交換熱量 [$\mathrm{kJ\,h^{-1}}$]
熱媒 1	330.0	11735.8
熱媒 2	55.0	1085.7
熱媒 3	2.3	1761.1

図 9.1 に最終的な設計結果における TQ 線図を示す.

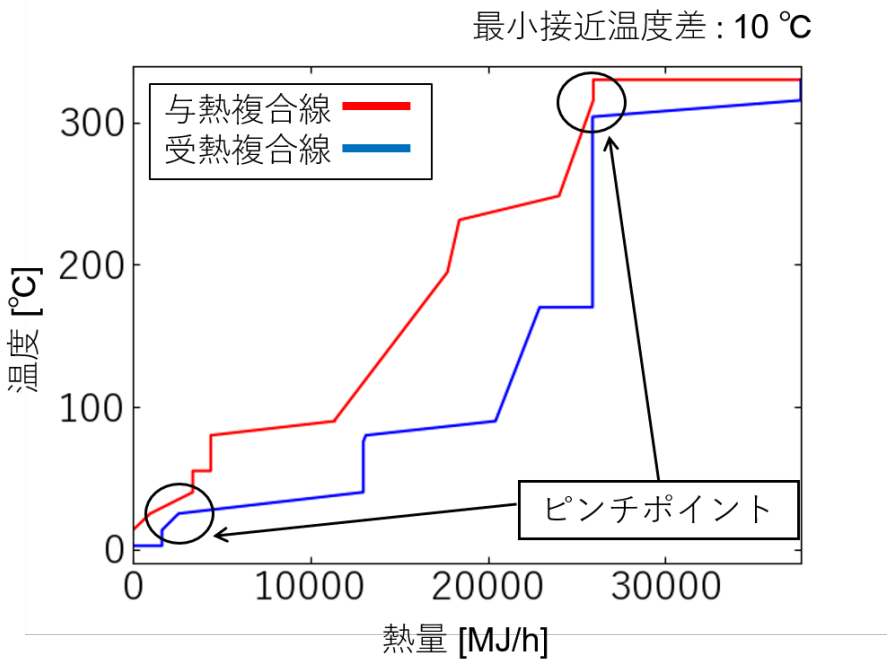

図 9.1　TQ 線図

ヒートインテグレーションを行うことで，必要な外部熱媒の費用は 3.6 億円から 1.2 億円にまで削減することができた.

第 10 章

経済評価

減価償却期間を 8 年とした場合の経済評価は表 10.1 のようになった．なお，各装置費の推算式は，Appendix A に示した通りである．また，市場調査により，トルエンは 500 $/t，安息香酸は 1013 $/t とした．

表 10.1　経済評価 [億円/年]

収入	製品	安息香酸	23.7	23.7
支出	原料コスト	トルエン	10.4	10.4
	装置コスト	晶析器	0.39	1.42
		反応器	0.34	
		熱交換器	0.33	
		抽出塔	0.14	
		コンデンサ	0.05	
		燃焼炉	0.05	
		蒸留塔	0.04	
		デカンタ	0.04	
		ポンプ	0.02	
		コンプレッサー	0.02	
	運転コスト	用役コスト	1.24	6.60
		触媒	0.36	
		人件費	5	
	総コスト			18.42
収益				5.28

第 11 章

結言

設計目標を純度 99.0 wt% の安息香酸を年 2 万 ton 製造するものとして，設計を行った．文献値を参考として主に気液反応器と晶析装置を設計した．気液反応器については，装置形状や内部流体の物性を考慮したうえで，気液間物質移動を含めた収支式を用いて詳細な解析を行った．また，晶析装置については各速度過程を考慮して解析を行った．リサイクルフローを多く設置したプロセスを考えたことに加えて，蒸発したトルエンを燃焼炉で燃やし，燃焼熱を利用するようなプロセスを考えたため，原料を最大限に活用することができた．それに伴い，環境に十分配慮したプロセスとなった．プロセスを最適化するにあたり，反応器体積，晶析器体積，晶析器内温度の 3 変数を同時に最適化することで，プロセス全体を考慮した上での最適値を得ることができた．結果として年間 5.39 億円の利益を見込めるプロセスとなった．

残った課題としては，蒸発するトルエンのさらなる回収を行うため，燃焼炉ではなく吸着装置を用いるプロセスと比較検討することや，晶析装置について結晶粒系分布を考慮すること，また，抽出塔について詳細な設計を行うことが挙げられる．さらに，熱交換構成について具体的な熱交換器の形状およびシステムを考慮する余地がある．本プロセスはダウ法フェノール製造工程の安息香酸生成部分を参考に設計を行ったため，安息香酸からフェノールの製造プロセスを検討することも可能である．

謝辞

今回のプロセス設計では，様々な方々にお世話になりました．山本教授，谷口准教授をはじめとする多くの化学工学の先生方，集中講義を実施してくださった玉川先生に感謝の意を申し上げます．

また，1 講座の先輩方の協力無しには私たちのプロセス設計は完了できませんでした．プロセスの内容や発表に対し，的確なアドバイスをいただきました．さらに，お忙しい中，原稿やスライドのチェック，発表練習などに協力して頂いたことで，無事に発表を終えることができました．

私たちのプロセス設計に協力してくださった先生方，先輩方に改めて御礼申し上げます．

参考文献

[1] Hexa Research, Benzoic Acid Market Analysis, 2016 https://www.hexaresearch.com/research-report/benzoic-acid-market

[2] J.A.A. Hoorn and J. Van Soolingen and G.F. Versteeg, *Chemical Engineering Research and Design*, **83**, 187 - 195, 2005

[3] 化学工学会・化学工学便覧・丸善出版

[4] 冨山 明男, 片岡 勲 and 坂口 忠司, 日本機械学会論文集 *B* 編, **61**, 2357-2364, 1995

[5] Wilke C.R. and P.Chang, *AIChE J.*, **1**, 264, 1955

[6] 京都大学化学プロセス工学コース 実験テキスト

[7] Jasper, J.J. The surface tension of pure liquid compounds, *J. Phys. Chem. Ref. Data*, **1**, 841 - 1010, 1972

[8] R.Byron Bird and Warren E. Stewart, Edwin N. Lightfoot TransPortPhenomena revised second edition willy

[9] Gary Morris and Graham Power, *et al. Org. Process Res. Dev.*, **19**, 1891-1902, 2015

[10] 東京都福祉保健局 http://www.fukushihoken.metro.tokyo.jp/kankyo/kankyo_eisei/jukankyo/indoor/sickhouse_faq/sick_faq_04.html

[11] Li, Wang and Zhang, Qingjun and Zeng, Aiwu, *Transactions of Tianjin University*, **25**, 52-65, 2019,

[12] 富士フィルム和光純薬 (株) https://labchem-wako.fujifilm.com/jp/product/detail/W01W0113-0065.html

[13] キシダ化学 (株) http://www.kishida.co.jp/product/catalog/detail/id/960

[14] 化学工学会・SIS 部会・情報技術教育分科会, 第 10 回プロセスデザイン学生コンテスト

[15] 京都大学・プロセス設計講義資料 3

[16] 第 6 回ソフトウェアツール学生コンテスト http://www.chemeng.titech.ac.jp/~sis_cont/dairokkai.html

[17] Gizli, Ali and Aytimur, Guelin and Alpay, Erden and Atalay, *Chemical Engineering & Technology*, **31**, 2008

[18] Versteeg, G and L. Alsters, P and A. A. Hoorn, J, *International Journal of Chemical Reactor Engineering*, **3**, 01, 2005

[19] Tang and Liang, Bin, *Industrial & Engineering Chemistry Research*, **46**, 6442-6448, 2007

[20] Ståhl, Marie and Åslund, Bengt L. and Rasmuson, Åke, *AIChE Journal*, **47**, 1544-1560, 2001

変数一覧

a:比界面積
B^0:一次核発生速度
c:重量濃度
C:モル濃度
C_{D}:抗力係数
ΔC:過飽和度
D:拡散係数
d_{vs}:気泡体積平均径
E:活性化エネルギー
E_{g}:結晶化過程の活性化エネルギー
F:モル流量
g:重力加速度
G:核成長速度
k:反応速度定数
k_{b}:核発生速度定数
k_{g}:核成長速度定数
k_{L}:液相物質移動係数
$k_{\mathrm{L}}a$:液相物質移動容量係数
k_{v}:結晶体積形状係数
M_{T}:懸濁密度
n:結晶の個数密度
n^0:$=B^0/G$
P_{G}:撹拌動力
P_{T}:
r:反応速度
T:温度
T_{c}:臨界温度
u_{t}:気泡の終末速度
u_{G}:気泡の空塔速度
V:体積
β:還流率
μ:粘度
μ_{L}:液相粘度
σ:界面張力
ρ_{a}:空気密度
ρ_{L}:液相密度
ρ_{g}:気相密度
τ:空間時間
$\Delta\rho$:密度差
Eo:エトベス数
Sc:シュミット数
Re:レイノルズ数

付録 A

コスト推算

為替レートは 1 ドル=111.73 円とした．(2019 年 4 月平均)

A.1　労務費

労務費の推算について補足する．プラントは 4 直 3 交代で運転され，1 班の人数に関して次の推算式を用いた．

$$(1\text{ 班の人数}) = (6.29 + 0.23 \times (\text{主要機器数}))^{0.5} \tag{A.1}$$

よって，総運転員数は 40 人であり，主任などに 10 人加え，50 人に平均 1000 万円の給与を支払うとして労務費を算出した．

A.2　ユーティリティコスト

用役単価を表 A.1 に示す．触媒であるモリブデン酸マンガンは材料である酢酸マンガン 21 g とモリブデン

表 A.1　用役単価

項目	価格
燃料 [$/ GJ]	1.095
触媒 [円/g]	9.616
2.3 ℃プロピレン冷媒 [$/ GJ]	4.804
純水 [$/ t]	45
電力 [$/ kW h]	0.1

酸アンモニウム 5.04 g を混ぜて作られる [11]．各価格について酢酸マンガンは 500 g あたり 3100 円とし ([12], 2019/7/17 確認)，モリブデン酸アンモニウムは 500 g あたり 11900 円 ([13]，2019/7/17 確認) とした．燃料，プロピレン冷媒 (内挿値) の価格は参考文献 [14] より得た．電力の価格は参考文献 [15] から得た．

さらに排水処理費用としてデカンターからパージする純水に 1 t あたり 0.041 $かかるとした [15].

A.3　機器コスト

下記の推算は 2001 年のデータで行い，2001 年のコストインデックス 394 と 2018 年のコストインデックス 603.1 を用いて値を修正している [15]．主要機器について，

1) 常圧で運転することを想定し，炭素鋼を用いて作成されるとしてメーカー出荷地点での価格を推算

2) 関連する部分の直接費，間接費を含めた価格の推算

3) 圧力や材質利用に関する補正

という 3 段階によって機器の建設費を推定する方法を用いた．

まず，メーカー船積み出荷価格 C_{p}^{0} は，機器の特徴サイズ A と係数 K_1, K_2, K_3 を用いて

$$\log_{10} C_{\mathrm{p}}^{0} = K_1 + K_2 \log_{10} A + K_3 (\log_{10} A)^2 \tag{A.2}$$

直接費や間接費，特殊材料費，操作圧力を考慮すると，各装置に関係するコストは C_{p}^{0} の数倍になる．すなわち，

$$C_{\mathrm{BM}} = F_{\mathrm{BM}} C_{\mathrm{p}}^{0} \tag{A.3}$$

と表される．C_{BM}, F_{BM} はそれぞれベアモジュールコスト，ベアモジュールファクターと呼ばれる．F_{BM} は，

$$F_{\mathrm{BM}} = B_1 + B_2 F_{\mathrm{p}} F_{\mathrm{M}} \tag{A.4}$$

と表される．ここで，B_1, B_2, F_{p}, F_{M} はそれぞれ圧力，材質に依存しない部分の係数，依存する部分の係数，圧力ファクター，材質ファクターを表す．F_{p} に関しては槽型の装置に対して推算式 (A.5) が提示されている．

$$F_{\mathrm{p,vessel}} = \begin{cases} \max\left\{ \dfrac{(P_{\mathrm{g}}+1)D}{10.71 - 0.00756(P_{\mathrm{g}}+1)} + 0.5\,,\, 1 \right\} & (P_{\mathrm{g}} > -0.5\,\mathrm{bar}) \\ 1.25 & (P_{\mathrm{g}} \le -0.5\,\mathrm{bar}) \end{cases} \tag{A.5}$$

ただし，P_{g} はゲージ圧 [bar] である．槽型以外の装置については，次式を用いた．

$$\log_{10} F_{\mathrm{p}} = C_1 + C_2 \log_{10} P_{\mathrm{g}} + C_3 (\log_{10} P_{\mathrm{g}})^2 \tag{A.6}$$

各機器のコスト算出に当たって用いた係数を表 A.2 に示す．データが資料にない機器についてはベアモジュールファクターを 1 とした．

表 A.2　ベアモジュールファクター算出に用いた係数

	A	K_1	K_2	K_3	B_1	B_2	C_1	C_2	C_3	材質	F_{M}
反応器	体積 [m^3]	4.5587	0.2986	0.002	1.49	1.52	-	-	-	Ti clad	4.8
晶析器	体積 [m^3]	4.5097	0.1781	0.1344	1.49	1.52	-	-	-	Ti alloy clad	9.4
蒸留塔 (槽)	体積 [m^3]	3.4974	0.4485	0.1074	1.49	1.52	-	-	-	Ti clad	4.8
蒸留塔 (トレイ)	面積 [m^2]	2.9949	0.4465	0.3961	1.49	1.52	-	-	-	Ti clad	4.8
抽出塔	体積 [m^3]	3.4974	0.4485	0.1074	1.49	1.52	-	-	-	Ti	9.4
デカンター	体積 [m^3]	3.4974	0.4485	0.1074	2.25	1.82	-	-	-	CC	1
燃焼炉	燃焼熱量 [kW]	3.068	0.6597	0.0194	-	-	0	0	0	CC	1
ポンプ	電力 [kW]	3.8696	0.3161	0.122	1.89	1.35	0	0	0	Ni alloy	3.9
コンプレッサー	電力 [kW]	2.2897	1.3604	-0.103	-	-	0	0	0	CC	1

熱交換器については，伝熱面積を $A\,[\mathrm{m}^2]$ として次の計算式を用いた [16]．

$$(\text{コスト [億円]}) = 0.015K \times A^{0.65} \tag{A.7}$$

ただし，K は定数でありコンデンサーは 1，リボイラーは 2 とした．

付録 B

プログラム

B.1 python

気液反応器の解析をします．特に断りがなければ SI 単位です FlowNO; 0=反応器への流入液,1=反応器からの流出液,2=反応器への流入空気， 3=反応からデカンターへの蒸気 4=反応器への還流油分,5=デカンターからの流出空気， 6=デカンターに入れる冷却水,7=デカンターからの流出水 ComponentNO; 0=Tol,1=Bol,2=Bal,3=Bac,4=H2O,5=N2,6=O2

import math as mt import numpy as np import matplotlib.pyplot as plt import openpyxl as px

————————外部入力値————————————————————- 結果出力用ファイル設定 (ソースファイルディレクトリ) conclusionbookname = ”conclusion.xlsx”

0. 外部入力条件を受けとります液相体積 VL = 0.020 流入液体積 [m3/s] v0 = 0.020/3600. 反応器体積 DHrate = 1.0 全圧 Prea = 7.0e+5 温度 Trea = 150. + 273.15 流入液モル分率 x0 = np.array([0.9, 0., 0., 0., 0.1, 0., 0.]) 流入液全モル量 [mol/s] F0all = 0.005

———————————————————————— 1 入力 1-1 計算に用いる物性パラメータを入力します自然定数 pi = mt.pi Rgas = 8.314 gra = 9.81

アレニウス型反応速度式のパラメータ頻度因子 [s-1] A = np.array([mt.exp(20.634), mt.exp(17.928), mt.exp(15.4), mt.exp(19.698)]) A = A / 3600. 活性化エネルギー E = np.array([81.389, 69.53, 56.987, 71.442]) E = E * 1000. 反応速度定数 [s-1] k = np.array([A[i] * mt.exp(-E[i] / (Rgas * Trea)) for i in range(4)])

物性値 (HYSIS 150 ℃) rhoL = 738.6 muL = 0.1781 / 1000. rhoG = 5.74

物性値 (推算) 気相から液相への拡散係数 wilke-chang の式とアインシュタイン-ストークスの式によって計算 DL = 0.0001 * 0.000000074 * ((1. * 92.141) ** 0.5 * 293.15) / (0.579 * (29.9 ** 0.6)) DL = (Trea / 293.15) * ((0.579 / 1000.) / muL) * DL

表面張力 温度依存性の表式でデータを換算 sig = 0.0221 * ((591.7 - Trea) / (591.7 - 353.15)) ** 1.24

1-2 反応器の装置条件を決めます液相部高さ 反応器直径，断面積 Drea = (VL / (pi * DHrate)) ** (1 / 3) hL = Drea * DHrate Area = pi * (Drea ** 2) ノズル直径，断面積 Dnoz = Drea / 1.5 Anoz = pi * (Dnoz ** 2)

1-3 反応器の運転条件を決めます空間時間 tau = VL / v0 体積当たり撹拌動力 Pv = 2000. 反応器への空気体積流量 QG = (50. / 1000. / 3600.) * (VL / 0.00025) ** 0.8 デカンター内温度 Tdec = 50. + 273.15 気体の空塔速度 uG = QG / Area 気体の投入線速度 u = QG / Anoz 撹拌動力 PG = Pv * VL 気体の流入運動エネルギー PK = 0.5 * QG * rhoG * u ** 2 気体の流入位置エネルギー Pg = QG * rhoL * gra * hL 総エネルギー PT = PG + PK + Pg

1-3 マテリアルフローに必要な変数を入力します

T2 = 20. + 273.15

各フローのモル分率 x0 x1 = np.array(np.zeros(7)) y2 = np.array([0, 0, 0, 0, 0, 0.79, 0.21]) y3 = np.array(np.zeros(7)) y4 = np.array([1., 0, 0, 0, 0, 0, 0]) y5 = np.array([0., 0., 0., 0., 0., 0., 0.]) y6 = np.array([0., 0., 0., 0., 1., 0., 0.]) y7 = np.array([0, 0, 0, 0, 1., 0, 0])

各フローのモル流量 F0 = np.array(x0 * F0all) F1 = np.array([0.] * 7) F2 = np.array([0., 0., 0., 0., 0., ((Prea * 0.79) / (Rgas * T2)), ((Prea * 0.21) / (Rgas * T2))]) F2 = np.array(F2 * QG) F3 = np.array([0.] * 7) F4 = np.array([0.] * 7) F5 = np.array([0.] * 7) F6 = np.array([0.] * 7) F7 = np.array([0.] * 7)

各フローのモル濃度 C0 = np.array(F0 / v0) C1 = np.array([0.] * 7)

---

2 kLa の推算を行います

体積平均気泡径と終末速度に関して逐次代入計算を行います．気泡径と終末速度の初期値です．dvs = 0. ut = 0. dvsnew = 0.003 epsG = 0. Eo = 0. Re = 0.

Eo はエトベス数，Re は気泡レイノルズ数，CD は抗力係数 while abs((dvs - dvsnew) / dvsnew) ¿ 0.00001: dvs = dvsnew 定義 Eo = gra * (rhoL - rhoG) * dvs / sig

utnew = 0.15 while abs(((ut - utnew) / utnew)) ¿ 0.00001: ut = utnew 定義 Re = dvs * ut * rhoL / muL 抗力係数の相関 CD = max((24 / Re * (1 + 0.15 * Re ** 0.687)), (8 / 3 * Eo / (Eo + 4))) 気泡の運動方程式 utnew = (4 / 3 / CD * (1 - rhoG / rhoL) * gra * dvs) ** 0.5

ut = utnew ガスホールドアップの相関式 (二次方程式) Const1 = -(uG / ut) ** 0.5 Const2 = -0.000216 * (Pv ** 0.4 * rhoL ** 0.2 / sig ** 0.6) * ((uG / ut) ** 0.5) * (PT / PG) epsG = (0.5 * (-Const1 + mt.sqrt(Const1 ** 2 - 4 * Const2))) ** 2 気泡径の相関 dvsnew = 4.15 * (sig ** 0.6 / (Pv ** 0.4 * rhoL ** 0.2)) * (PG / PT) * epsG ** 0.5 + 0.0009 dvs = dvsnew

反応器内物質高さ，体積 (液相 + 気相) htot = hL / (1 - epsG) Vtot = htot * Area 体積当たり比表面積 a の相関 a = 1.44 * ((Pv ** 0.4 * rhoL ** 0.2) / sig ** 0.6) * (PT / PG) * (uG / ut) ** 0.5

液相側物質移動係数 kL の相関気泡径 0.6mm 以下の時 if dvs ¡= 0.0006: kL = 0.31 * ((muL / (rhoL * DL)) ** (-2 / 3)) * ((gra * (rhoL - rhoG) * muL / (rhoL ** 2)) ** (1 / 3))

気泡系 0.6mm 以上 2.5mm 以下の時 elif 0.0006 ¡ dvs ¡= 0.0025: x = (dvs - 0.0006) / (0.0025 - 0.0006) kL = (1 - x) * 0.31 * ((muL / (rhoL * DL)) ** (-2 / 3)) * ((gra * (rhoL - rhoG) * muL / (rhoL ** 2)) ** (1 / 3)) + x * (0.42 * ((muL / (rhoL * DL)) ** (-0.5)) * ((gra * (rhoL - rhoG) * muL / (rhoL ** 2)) ** (1 / 3)))

気泡径 2.5mm 以上の時 else: kL = 0.42 * ((muL / (rhoL * DL)) ** (-0.5)) * ((gra * (rhoL - rhoG) * muL / (rhoL ** 2)) ** (1 / 3))

液相物質移動容量係数 kLa kLa = kL * a

---

3 液相の油分について，蒸発の影響を無視小，CSTR モデルと仮定して反応器内濃度を求めます

tol C1[0] = C0[0] / (1 + (k[0] + k[1]) * tau) Bol C1[1] = (C0[1] + k[0] * tau * C1[0]) / (1 + k[2] * tau) Bal C1[2] = (C0[2] + (k[1] * C1[0] + k[2] * C1[1]) * tau) / (1 + k[3] * tau) Bac C1[3] = C0[3] + k[3] * tau * C1[2]

各反応の反応量 Rr = np.array([k[0] * C1[0], k[1] * C1[0], k[2] * C1[1], k[3] * C1[2]]) Rr = Rr * VL 各成分の反応量 RR = np.array([-(Rr[0] + Rr[1]), Rr[0] - Rr[2], Rr[1] + Rr[2] - Rr[3], Rr[3], Rr[1] + Rr[2], 0., -(0.5 * Rr[0] + Rr[1] + 0.5 * Rr[2] + 0.5 * Rr[3])]) print(C0) print(C1) print(RR)

---

4 気相について，RK4 法を用いて蒸発量を推算します 4-first 計算に用いる関数を定義

def Gaseq(Fgas): 気相側の物質収支式

def Csatcalc(yg): 平衡濃度 Csat を求める関数 Cs = np.array([0.] * 7) xe = np.array([0.] * 7)

tol,H2O=ラウール則,N2,O2=ヘンリー則で計算，残りは蒸発しないとする xe[0] = Prea * yg[0] / 10 ** (Aant[0] - Bant[0] / (Trea + Cant[0])) xe[1] = x1[1] xe[2] = x1[2] xe[3] = x1[3] xe[4] = Prea * yg[4] / mt.exp(Aant[4] - Bant[4] / (Trea + Cant[4])) xe[5] = Prea * yg[5] / HenryN2 xe[6] = Prea * yg[6] / HenryO2

Cs = sum(C1) * xe return (Cs)

定義 Csat = np.array([0.] * 7) dFgdz = np.array([0.] * 7) ygas = np.array([0.] * 7)

気相モル分率 y を求め，平衡濃度 Csat を算出 ygas = Molefraction(Fgas) Csat = Csatcalc(ygas)

収支式の計算 const = kLa * pi * Drea ** 2 dFgdz = np.array(-const * (Csat - C1))

return (dFgdz)

def Molefraction(Flow): fraction = np.array(Flow / sum(Flow)) return (fraction)

4-1 計算に用いる気液平衡データを入力

アントワン定数 10** or exp(A-B/(T+C)) tol=10** H2O=exp Aant = np.array([4.54436 + 5.0, 0., 0., 0., 23.1964, 0., 0.]) Bant = np.array([1738.123, 0., 0., 0., 3816.44, 0., 0.]) Cant = np.array([0.394, 0., 0., 0., -46.13, 0., 0.])

ヘンリー定数 pall*y=Henry*x HenryN2 = 187000000. HenryO2 = 114000000.

4-2 rk4 法を用いて気相の物質収支式を解き，全体の物質収支と照らし合わせて収束させる

窒素濃度の初期値．これは蒸発も溶解もしないように収束させる C1[5] = 0.5 水と酸素の初期値．これらは反応生成量，消費量が蒸発 (溶解) 量と等しくなるように収束させる C1[4] = 10. C1[6] = 0.5

誤差評価関数初期値 eps = 1000.

ステップ数 step = 50 刻み幅 h = htot / step

j = 0 while eps ¿= 0.001 and j ¡= 100: j = j + 1 x1 = Molefraction(C1) 計算用 Fg = F2.copy()

for i in range(0, step - 1):

kk1 = np.array(h * Gaseq(Fg)) kk2 = np.array(h * Gaseq(Fg + kk1 / 2.)) kk3 = np.array(h * Gaseq(Fg + kk2 / 2.)) kk4 = np.array(h * Gaseq(Fg + kk3))

Fg += (kk1 + 2. * (kk2 + kk3) + kk4) / 6. next

結果 F3 = Fg.copy()

誤差を求める epslist = np.array([((F3[4] - RR[4]) / RR[4]) ** 2, ((F2[5] - F3[5]) / F2[5]) ** 2, (((F3[6] - F2[6]) - RR[6]) / RR[6]) ** 2]) eps = sum(epslist)

print(eps)

値の改善

C1[4] += - (F3[4] - RR[4]) C1[5] += (F2[5] - F3[5]) C1[6] += - ((F3[6] - F2[6]) - RR[6])

4-3 収束時の分布を取得

---

5 デカンター周りの各流量を求め，液相の反応に対して蒸発の影響が無視できるか確認する 5-1 各流量を算出する．F5 について，デカンターで Tdec まで蒸気 F3 が冷やされ，水とトルエンが飽和状態まで凝縮する．y3 = Molefraction(F3) F5 = F3.copy() y5sat = np.array([10 ** (Aant[0] - Bant[0] / (Tdec + Cant[0])) / Prea, 0, 0, 0, mt.exp(Aant[4] - Bant[4] / (Tdec + Cant[4])) / Prea, 0, 0]) F5[0] = y5sat[0] / (1 - y5sat[0] - y5sat[4]) * (F5[5] + F5[6]) F5[4] = y5sat[4] / (1 - y5sat[0] - y5sat[4]) * (F5[5] + F5[6]) y5 = Molefraction(F5)

溶解度が低いため，デカンターにおける分離効率は近似的に 100F4[0] = F3[0] - F5[0]

冷却のため投入する水量 F6 を計算する

F7 を計算 F7[4] = F6[4] + F3[4] - F5[4]

6 結果を出力する

wb = px.load$_{w}orkbook(conclusionbookname)ws = wb.active$

for i in range(7): ws.cell(row=i + 2, column=2).value = F2[i] ws.cell(row=i + 2, column=3).value = F3[i] ws.cell(row=i + 2, column=4).value = F4[i] ws.cell(row=i + 2, column=5).value = F5[i]

ws.cell(row=i + 10, column=1).value = x1[i] ws.cell(row=i + 10, column=2).value = y2[i] ws.cell(row=i + 10, column=3).value = y3[i] ws.cell(row=i + 10, column=4).value = y4[i] ws.cell(row=i + 10, column=5).value = y5[i]

wb.save(filename=conclusionbookname)

迅速に気相が平衡に達すると仮定した場合の上記プログラムは以下のようになり，実際の解析ではこちらを用いた． 気液反応器の解析をします．特に断りがなければ SI 単位です

FlowNO：0=反応器への流入液, 1=反応器からの流出液, 2=反応器への流入空気, 3=反応からデカンターへの蒸気 4=反応器への還流油分, 5=デカンターからの流出空気， 6=デカンターに入れる冷却水, 7=デカンターからの流出水 ComponentNO: 0=Tol, 1=Bol, 2=Bal, 3=Bac, 4=H2O, 5=N2, 6=O2 ver$_{s}lim$: 迅速な平衡の仮定から各流量を求める $slim$ な $program$ です $importmathimportnumpyasnpimportxlwingsasxw$

def main():

——————Excel シートから data を取得—————————— wb = xw.Book.caller() F0$_{i}nput = wb.sheets[0].range((43,4),(43,10)).value$ 反応器入口成分別モル流量 $P_{i}nput = wb.sheets[0].range('D3').value$ 反応器圧力 $VL_{i}nput = wb.sheets[0].range('D4').value$ 反応器体積 $T_{i}nput = wb.sheets[0].range('D5').value$ 反応器温度 $T_{d}ecanter = wb.sheets[0].range('D7').value$ デカンタ温度 $v0_{i}nput = wb.sheets[0].range('N43').value$ 反応器入口体積流量 $topRow = 46 --$

流れのクラスと流れに関係するメソッド class Flow:

中身 def $_{init}{}_{(self):}$モル流量，モル分率，濃度，平衡での気液モル分率，温度を持っています $self.MolerFlow=np.array([0.]*7)self.Fraction=np.array([0.]*7)self.Concentration=np.a$

アントワン定数 10** or exp(A-B/(T+C)) tol=10** H2O=exp antoineA = np.array([4.54436 + 5.0, 0., 0., 0., 23.1964, 0., 0.]) antoineB = np.array([1738.123, 0., 0., 0., 3816.44, 0., 0.]) antoineC = np.array([0.394, 0., 0., 0., -46.13, 0., 0.])

ヘンリー定数 pall*y=Henry*x HenryN2 = 187000000. HenryO2 = 114000000.

標準モル生成エンタルピー LiquidDeltafEnthalpy = np.array([12.4 ,-94.1 ,-87.0 ,-385.2 ,-285.83 ,0. ,0.]) GasDeltafEnthalpy = np.array([50.4 ,154.9 ,-36.8 ,-290.4 ,-241.818 ,0. ,0.])

熱容量係数 GasheatcapacityA = np.array([-24.356 , 0. , 0. ,-51.295 ,32.244 ,28.107 ,24.234]) GasheatcapacityB = np.array([0.513 , 0. , 0. , 0.629 ,1.92e-3 ,-3.68e-6 ,4.84e-3]) GasheatcapacityC = np.array([-2.77e-5 , 0. , 0. , -4.24e-5 ,1.06e-5 ,1.75e-5 ,-2.08e-5]) GasheatcapacityD = np.array([4.91e-8 , 0. , 0. , 1.06e-7 ,-3.60e-9 ,-1.07e-8 ,2.93e-10]) 熱容量係数 LiqheatcapacityA = np.array([157.09 , 0. , 0. ,0. ,75.375 , 0. , 0.]) LiqheatcapacityB = np.array([0. , 0. , 0. ,0. ,0. , 0. , 0.]) LiqheatcapacityC = np.array([0. , 0. , 0. ,0. ,0. , 0. , 0.]) LiqheatcapacityD = np.array([0. , 0. , 0. ,0. ,0. , 0. , 0.])

モル流量からモル分率を求めるメソッド def FractionCalc(self): FlowSum = np.sum(self.MolerFlow) self.Fraction = self.MolerFlow/FlowSum

濃度からモル分率を求めるメソッド

def FractionCalcFromConcentration(self): FlowSum = np.sum(self.Concentration) self.Fraction = self.Concentration/FlowSum

decanter で Tol と水の平衡状態の気体モル分率を求めるメソッド

def DecanterEquilibriumGasFractionCalc(self): self.EquilibriumGasFraction[0] = (10 ** (self.antoineA[0] - self.antoineB[0] / (Tdec + self.antoineC[0])))/Prea self.EquilibriumGasFraction[4] = (math.exp(self.antoineA[4] - self.antoineB[4] / (Tdec + self.antoineC[4])))/Prea

CSTR(液相) の設計方程式から油分の液相濃度を求めさせるメソッド def CSTRLiquidCalc(self): tol Flow1.Concentration[0] = Flow0.Concentration[0] / (1 + (k[0] + k[1]) * tau) Balc Flow1.Concentration[1] = (Flow0.Concentration[1] + k[0] * tau * Flow1.Concentration[0]) / (1 + k[2] * tau) Bald Flow1.Concentration[2] = (Flow0.Concentration[2] + (k[1] * Flow1.Concentration[0] + k[2] * Flow1.Concentration[1]) * tau) / (1 + k[3] * tau) BzA Flow1.Concentration[3] = Flow0.Concentration[3] + k[3] * tau * Flow1.Concentration[2]

reactor で平衡状態の液モル分率を求めるメソッド def EquilibriumLiquidFractionCalc(self): self.EquilibriumLiquidFraction[0] = Prea * self.Fraction[0] / 10 ** (self.antoineA[0] - self.antoineB[0] / (Trea + self.antoineC[0])) self.EquilibriumLiquidFraction[1] = Flow1.Fraction[1] self.EquilibriumLiquidFraction[2] = Flow1.Fraction[2] self.EquilibriumLiquidFraction[3] = Flow1.Fraction[3] self.EquilibriumLiquidFraction[4] = Prea * self.Fraction[4] / math.exp(self.antoineA[4] - self.antoineB[4] / (Trea + self.antoineC[4])) self.EquilibriumLiquidFraction[5] = Prea * self.Fraction[5] / self.HenryN2 self.EquilibriumLiquidFraction[6] = Prea * self.Fraction[6] / self.HenryO2

気流が迅速に平衡に達すると仮定してほかの流量や濃度を求めるメソッド def EquilibriumAssumption-Calc(self):

Flow1.FractionCalcFromConcentration()

F2 から F3 の決定 Flow3.MolerFlow[4] = RR[4] Flow3.MolerFlow[5] = Flow2.MolerFlow[5] Flow3.MolerFlow[6] = Flow2.MolerFlow[6] + RR[6]

Flow3.Fraction[0] = Flow1.Fraction[0] / Prea * 10. ** (self.antoineA[0] - self.antoineB[0] / (Trea + self.antoineC[0])) sumF3 = np.sum(Flow3.MolerFlow) Flow3.MolerFlow[0] = Flow3.Fraction[0] / (1-Flow3.Fraction[0])*(sumF3) Flow3.FractionCalc()

迅速に平衡に達すると仮定して F1 の H2O,N2,O2 濃度を決定 Flow3.EquilibriumLiquidFractionCalc() C1sum=np.sum(Flow1.Concentration) Flow1.Fraction[4] = Flow3.EquilibriumLiquidFraction[4] Flow1.Fraction[5] = Flow3.EquilibriumLiquidFraction[5] Flow1.Fraction[6] = Flow3.EquilibriumLiquidFraction[6] Flow1.Concentration[4] = Flow1.Fraction[4] / (1-Flow1.Fraction[4]) * C1sum Flow1.Concentration[5] = Flow1.Fraction[5] / (1-Flow1.Fraction[5]) * C1sum Flow1.Concentration[6] = Flow1.Fraction[6] / (1-Flow1.Fraction[6]) * C1sum Flow1.FractionCalcFromConcentration() Flow1.MolerFlow = Flow1.Concentration * v0

F5 について，デカンターで Tdec まで蒸気 F3 が冷やされ， 水とトルエンが飽和状態まで凝縮するとして決定 Flow5.MolerFlow = np.copy(Flow3.MolerFlow) Flow5.DecanterEquilibriumGasFractionCalc() Flow5.MolerFlow[0] = Flow5.EquilibriumGasFraction[0] / (1 - Flow5.EquilibriumGasFraction[0] - Flow5.EquilibriumGasFraction[4]) * (Flow5.MolerFlow[5] + Flow5.MolerFlow[6]) Flow5.MolerFlow[4] = Flow5.EquilibriumGasFraction[4] / (1 - Flow5.EquilibriumGasFraction[0] - Flow5.EquilibriumGasFraction[4]) * (Flow5.MolerFlow[5] + Flow5.MolerFlow[6]) Flow5.FractionCalc()

溶解度が低いため, デカンターにおける分離効率は近似的に 100Flow4.MolerFlow[0] = Flow3.MolerFlow[0] - Flow5.MolerFlow[0] F1 に蒸発の寄与を取り込む Flow1.MolerFlow[0] += -Flow5.MolerFlow[0]

冷却のため投入する水量 F6 を決定モルエンタルピーの計算 Flow1.LiquidEnthalpyCalc() Flow2.GasEnthalpyCalc() Flow3.GasEnthalpyCalc() Flow4.LiquidEnthalpyCalc() Flow5.GasEnthalpyCalc() Flow6.LiquidEnthalpyCalc() Flow7.LiquidEnthalpyCalc()

連立方程式を解き F6 と F7 を求める A1list=[1,-1] A2list=[Flow6.MolerEnthalpy[4] , -Flow7.MolerEnthalpy[4]] A$_matrix = np.array([A1list, A2list])b = np.array([Flow4.MolerFlow[4] + Flow5.MolerFlow[4] - Flow3.MolerFlow[4], Flow4.EnthalpyTotal + Flow5.EnthalpyTotal - Flow3.EnthalpyTotal])x_vec = np.linalg.solve(A_matrix, b)Flow6.MolerFlow[4] = x_vec[0]Flow7.MolerFlow[4] = x_vec[1]$

気体のモルエンタルピーを求めるメソッド [J/mol] def GasEnthalpyCalc(self): self.MolerEnthalpy = self.GasDeltafEnthalpy*1000+ self.GasheatcapacityA*(self.Temp -298.15) + self.GasheatcapacityB*(self.Temp**2.-298.15**2.)/2. + self.GasheatcapacityC*(self.Temp**3.-298.15**3.)/3. + self.GasheatcapacityD*(self.Temp**4.-298.15**4.)/4. self.Enthalpy = self.MolerFlow * self.MolerEnthalpy self.EnthalpyTotal=np.sum(self.Enthalpy)

液体のモル当たりエンタルピーを求めるメソッド [J/mol] def LiquidEnthalpyCalc(self): self.MolerEnthalpy = self.LiquidDeltafEnthalpy*1000 + self.LiqheatcapacityA*(self.Temp-298.15) self.Enthalpy = self.MolerFlow * self.MolerEnthalpy self.EnthalpyTotal=np.sum(self.Enthalpy)

インスタンスの定義 Flow0 = Flow() Flow1 = Flow() Flow2 = Flow() Flow3 = Flow() Flow4 = Flow() Flow5 = Flow() Flow6 = Flow() Flow7 = Flow()

————————外部入力値————————————————————

0. 外部入力条件を受けとります 液相体積 VL = VL$_input$ 流入液体積 $[m3/s]v0 = v0_input[m3/hour]v0 = v0/3600.$反応器体積 $DHrate = 1.0$ 全圧 $[Pa]Prea = P_input * 1.e + 5$ 温度 [℃]$Trea = T_input + 273.15$ 流入液モル流量 $[mol/s]Flow0.MolerFlow = np.array(F0_input)kmol/hourFlow0.MolerFlow = Flow0.MolerFlow/3.6mol/s$ −

1 入力 ———————————————————————— 1-1 計算に用いる物性パラメータを入力します 自然定数 pi = math.pi Rgas = 8.314 gra = 9.81

アレニウス型反応速度式のパラメータ 頻度因子 [s-1] A = np.array([math.exp(20.634), math.exp(17.928), math.exp(15.4), math.exp(19.698)]) A = A / 3600. 活性化エネルギー E = np.array([81.389, 69.53, 56.987, 71.442]) E = E * 1000. 反応速度定数 [s-1] k = np.array([A[i] * math.exp(-E[i] / (Rgas * Trea)) for i in range(4)])

1-2 反応器の装置条件を決めます Drea = (VL / (pi * DHrate)) ** (1 / 3) hL = Drea * DHrate Area = pi * (Drea ** 2)/4. Dnoz = Drea / 10 Anoz = pi * (Dnoz ** 2)/4.

1-3 反応器の運転条件を決めます tau = VL / v0 QG = (50. / 1000. / 3600.) * (VL / 0.00025)*0.6 Tdec = T$_decanter + 273.15$

global 宣言 kLa = 0.0 htot = 0.0 Rr = np.array([0.]*4) RR = np.array([0.]*7)

1-4 マテリアルフローに必要な変数を入力します

Twater = 25.0

Flow2.Temp = 10. + 273.15 Flow3.Temp = Trea Flow4.Temp = Tdec Flow5.Temp = Tdec Flow6.Temp = Twater + 273.15 Flow7.Temp = Tdec

各フローのモル分率 Flow0.FractionCalc() Flow2.Fraction = np.array([0, 0, 0, 0, 0, 0.79, 0.21]) Flow4.Fraction = np.array([1., 0, 0, 0, 0, 0, 0]) Flow6.Fraction = np.array([0., 0., 0., 0., 1., 0., 0.]) Flow7.Fraction = np.array([0, 0, 0, 0, 1., 0, 0])

各フローのモル流量 Flow2.MolerFlow = np.array([0., 0., 0., 0., 0., ((Prea * 0.79) / (Rgas * Flow2.Temp)), ((Prea * 0.21) / (Rgas * Flow2.Temp))]) Flow2.MolerFlow = Flow2.MolerFlow * QG

各フローのモル濃度 Flow0.Concentration = Flow0.MolerFlow / v0

2 液相の油分について，蒸発の影響を無視小， CSTR モデルと仮定して反応器内濃度を求めます ————————

Flow1.CSTRLiquidCalc()

各反応の反応量 Rr = np.array([k[0] * Flow1.Concentration[0], k[1] * Flow1.Concentration[0], k[2] * Flow1.Concentration[1], k[3] * Flow1.Concentration[2]]) Rr = np.array(Rr * VL) 各成分の反応量 RR = np.array([-(Rr[0] + Rr[1]), Rr[0] - Rr[2], Rr[1] + Rr[2] - Rr[3], Rr[3],Rr[1] + Rr[2], 0., -(0.5 * Rr[0] + Rr[1] + 0.5 * Rr[2] + 0.5 * Rr[3])])

3 気相について, 平衡に達すると仮定して F3 や他の流量をを推算します ———— Flow3.EquilibriumAssumptionCalc()

————————————出力————————————-

$H7_{6c}ooling = Flow6.MolerFlow[4]*(Flow7.MolerEnthalpy[4]-Flow6.MolerEnthalpy[4])/1000./1000.*3600H6_1000kmol_cooling = 1000.*1000./3600.*(Flow7.MolerEnthalpy[4]-Flow6.MolerEnthalpy[4])/1000./1000.*3600$

if $H7_{6c}ooling > H6_1000kmol_cooling : Flow6.MolerFlow[4] = 1000.*1000./3600.Flow7.MolerFlow[4] = Flow6.MolerFlow[4] - Flow5.MolerFlow[4] + Flow3.MolerFlow[4]$

Flow6.LiquidEnthalpyCalc() Flow7.LiquidEnthalpyCalc()

Flow6.Temp += -273.15 Flow7.Temp += -273.15 $F1_output$ = $Flow1.MolerFlow * 3.6$ 反応器出口成分別モル流量 $F2_output$ = $Flow2.MolerFlow * 3.6$ 反応器出口成分別モル流量 $F3_output$ = $Flow3.MolerFlow * 3.6$ 反応器出口成分別モル流量 $F4_output$ = $Flow4.MolerFlow * 3.6$ 反応器出口成分別モル流量 $F5_output$ = $Flow5.MolerFlow * 3.6$ 反応器出口成分別モル流量 $F6_output$ = $Flow6.MolerFlow * 3.6$ 反応器出口成分別モル流量 $F7_output$ = $Flow7.MolerFlow * 3.6$ 反応器出口成分別モル流量 $Flow1.EnthalpyTotal$ = $Flow1.EnthalpyTotal * 1.e-9Flow2.EnthalpyTotal$ = $Flow2.EnthalpyTotal*1.e-9Flow3.EnthalpyTotal = Flow3.EnthalpyTotal*1.e-9Flow4.EnthalpyTotal = Flow4.EnthalpyTotal*1.e-9Flow5.EnthalpyTotal = Flow5.EnthalpyTotal*1.e-9Flow6.EnthalpyTotal = Flow6.EnthalpyTotal * 1.e-9Flow7.EnthalpyTotal = Flow7.EnthalpyTotal * 1.e-9$

wb.sheets[0].range((44, 4), (44, 10)).value = $F1_output$ 流出液流量 $wb.sheets[0].range(44,12).value = Flow1.EnthalpyTotal$ 流出液流量

wb.sheets[0].range((topRow + 0, 4), (topRow + 0, 10)).value = $F2_output$ 流入空気流量 $wb.sheets[0].range((topRow+1,4),(topRow+1,10)).value = F3_output$ 反応器からデカンターへの蒸気 $wb.sheets[0].range((topRow+2,4),(topRow+2,10)).value = F4_output$ 反応器へ還流される油分 $wb.sheets[0].range((topRow+3,4),(topRow+3,10)).value = F5_output$ 吸着装置へ運ばれる Tol 飽和蒸気 $wb.sheets[0].range((topRow+4,4),(topRow+4,10)).value = F6_output$ 冷却のため投入する水量 $wb.sheets[0].range((topRow+5,4),(topRow+5,10)).value = F7_output$ デカンターからパージする水量 $wb.sheets[0].range((topRow+6,4),(topRow+6,10)).value = Flow1.Concentration$ 流出液流量

wb.sheets[0].range(topRow + 0, 15).value = Flow2.EnthalpyTotal 流入空気流量 wb.sheets[0].range(topRow + 1, 15).value = Flow3.EnthalpyTotal 反応器からデカンターへの蒸気 wb.sheets[0].range(topRow + 2, 15).value = Flow4.EnthalpyTotal 反応器へ還流される油分 wb.sheets[0].range(topRow + 3, 15).value = Flow5.EnthalpyTotal 吸着装置へ運ばれる Tol 飽和蒸気 wb.sheets[0].range(topRow + 4, 15).value = Flow6.EnthalpyTotal 冷却のため投入する水量 wb.sheets[0].range(topRow + 5, 15).value = Flow7.EnthalpyTotal デカンターからパージする水量

wb.sheets[0].range(74,6).value = $H7_{6c}oolingwb.sheets[0].range(74,7).value = Flow6.Tempwb.sheets[0].range(74,8).value = Flow7.Temp + 0.001$

wb.sheets[0].range(75,6).value = $H6_1000kmol_coolingwb.sheets[0].range(75,7).value = Twaterwb.sheets[0].range(75,8).value = T_decanter$

wb.sheets[0].range(76,6).value = $H6_1000kmol_coolingwb.sheets[0].range(76,7).value = T_decanterwb.sheets[0].range(76$

$Twater$

==========

プログラム: 熱交換の解析　ver010 ：グランドコンポジットカーブを描画します． ver020 ：追加 TQ 線図を描画します ver030 ：追加 各流体の熱交換対応を解析します ver040 ：追加 任意の等温外部熱媒の熱量計算を実装　使い方:　xlwings を import したエクセルファイルに，相状態 [気体，液体，凝縮ガス，蒸気液] = [”gas”,”liq”,”congas”,”vapliq”] を書き込む (一部データを入れていない総括伝熱係数の組み合わせがある) 等温流体には”give”,”receive”を明示する．(エラーを吐くときは人為的に微小な温度勾配をつける) 外部熱媒には”exgive”,”exreceive”と書く 横の列に交換熱量，熱交換前温度，熱交換後の温度の順に書き込む． 熱交換流体の総数を書き込む．(外部熱媒を含む) 最小接近温度差を決める． vba から Runpython コマンドで実行 .

==========

def main():

ライブラリ from scipy import interpolate from math import log import numpy as np import xlwings as xw import copy import sys from matplotlib import pyplot as plt from matplotlib.font$_manager import FontProperties from$

wb = xw.Book.caller()

==========外部入力値==========

readsheet = 0 読み込みたいシート NO(0 始まり) writesheet = 1 書きこみたいシート NO(0 始まり) row$_readstart$ = 64 読み込みたい表の左上の行 $column_readstart$ = 3 読み込みたい表の左上の列 $row_writestart$ = 2 書き込みたい表の左上の行 $column_writestart$ = 2 書き込みたい表の左上の列 ==========

sn$_input$ = $wb.sheets[readsheet].range((row_readstart - 1, column_readstart - 1)).valuemdt_input$ = $wb.sheets[readsheet].range((row_readstart, column_readstart - 1)).value$

streamnumber = int(sn$_input$) 熱交換流体の総数 $minimumdeltaTemperture = float(mdt_input)$ 最小接近温度差

sig = 1 切り捨てる少数点以下桁数

識別用授受関係 give = ”give” 与熱流体 receive = ”receive” 受熱流体 externalgive = ”exgive” 外部熱媒 externalreceive = ”exreceive” 外部冷媒 gives = [give ,externalgive] receives = [receive,externalreceive]

相状態関係 gas = ”gas” liquid = ”liq” condensategas = ”congas” vaporliquid = ”vapliq” phaselist = [gas, liquid, condensategas, vaporliquid]

twophaselist = [] for i in range (len(phaselist)): for j in range (len(phaselist)): if i ¡= j: twophaselist.append([phaselist[i], phaselist[j]]) next next

overall$_heattransfer_coefficient$ = 総括伝熱係数 $U[W/m2K][gasgas], [gasliq], [gascongas], [gasvapgas], [liqliq], [liqcong$ $[150., 200., 500., 500., 300., 1000., 1000., ”none”, 1500., ”none”]$

float 判定用 eps = 10.**(- float(sig))

熱媒検索用 streamindex$_give$ = $[]streamindex_receive = []streamindex_externalgive = []streamindex_externalreceive = []streamindex_externalheat = []$

class Heatstream():

def $_init_{(self,number):self.name=””ストリーム名\ self.property=””(giveorreceive)or(exgiveorexreceive)self.phase=””状態\ self.number=number\ ストリーム番号\ self.del}$

self.leftend$_externalheat_GCC$ = 0.0 外部熱媒用

def read(self): excel 連携用 self.name = wb.sheets[readsheet] .range((row$_readstart + self.number + 1, column_readstart + 0)).valueself.phase = wb.sheets[readsheet].range((row_readstart + self.number + 1, column_readstart+1)).valueself.property = wb.sheets[readsheet].range((row_readstart+self.number+1, column_readstart+2)).valueself.deltaheat = wb.sheets[readsheet].range((row_readstart+self.number+$

$1, column_readstart + 3)).valueself.temperatureMAX = wb.sheets[readsheet].range((row_readstart +$
$self.number+1, column_readstart+4)).valueself.temperatureMIN = wb.sheets[readsheet].range((row_readstart+$
$self.number+1, column_readstart+5)).valueself.rebcon = wb.sheets[readsheet].range((row_readstart+$
$self.number + 1, column_readstart + 6)).value$

ねんのため self.name = str(self.name) self.phase = str(self.phase) self.property = str(self.property) self.deltaheat = float(self.deltaheat) self.temperatureMAX = float(self.temperatureMAX) self.temperatureMIN = float(self.temperatureMIN)

ddmax = np.random.rand() * 0.0001 + 0.0001 等温熱媒調整用微小項 ddmin = np.random.rand() * 0.0001 + 0.0001 等温熱媒調整用微小項

if (self.temperatureMAX - self.temperatureMIN) ¿ 0 :与熱流体 self.property = give streamindex$_give.append(self.number$

elif (self.temperatureMAX - self.temperatureMIN) ¡ 0 :受熱流体 self.property = receive self.temperatureMAX, self.temperatureMIN = self.temperatureMIN, self.temperatureMAX streamindex$_receive.append(self.number)$

self.temperatureMAX += ddmax 傾きがないとエラーなので微小勾配をつける self.temperatureMIN += -ddmin

if self.property == externalgive :外部与熱流体 self.deltaheat = 0.0 streamindex$_externalgive.append(self.number)$

streamindex$_externalheat.append(self.number)$

elif self.property == externalreceive :外部受熱流体 self.deltaheat = 0.0 streamindex$_externalreceive.append(self.number$

streamindex$_externalheat.append(self.number)$

self.heatcapacity$_calc()$

def heatcapacity$_calc(self)$: $self.heatcapacity = self.deltaheat/(self.temperatureMAX - self.temperatureMIN)$

streams=[] 熱流体の情報読み込み for i in range (streamnumber): dummy = Heatstream(i) streams.append(dummy) streams[i].read()

ascendingtemperature$_TQ$ = [] 温度の昇順ソート $ascendingtemperature_GCC = []foriinrange(len(streams))$: $ascendingtemperature_TQ.append(streams[i].temperatureMAX)TQ$ 線図用 $ascendingtemperature_TQ.append(streams[$ $give : GCC$ 用 $ascendingtemperature_GCC.append(streams[i].temperatureMAX)ascendingtemperature_GCC.append(st$ $receive : ascendingtemperature_GCC.append(streams[i].temperatureMAX+minimumdeltaTemperture)ascendingtem$ $minimumdeltaTemperture)next$

ascendingtemperature$_TQ.sort()ascendingtemperature_GCC.sort()$

整理した温度の index を取得 for i in range (len(streams)): streams[i].temperatureindexMAX = ascendingtemperature$_TQ.index(streams[i].temperatureMAX)streams[i].temperatureindexMIN = ascendingtemperature_TQ.index(streams[i].temperatureMIN)$

heatbalance$_section_GCC = np.array([0.] * (len(ascendingtemperature_GCC)))$ 各温度範囲での熱バランス $heatbalance_sum_GCC = np.array([0.] * (len(ascendingtemperature_GCC)))GCC$ グラフ用の熱量

for i in range (len(ascendingtemperature$_GCC)-1) : GCC$ カーブデータの作成 $dT = (ascendingtemperature_GCC[i+$ $1]-ascendingtemperature_GCC[i])forjinrange(len(streamindex_give)) : x = streams[streamindex_give[j]]ifx.temperatu$ $ascendingtemperature_GCC[i] < x.temperatureMAX : heatbalance_section_GCC[i]+ = +x.heatcapacity*$ $dTforjinrange(len(streamindex_receive)) : x = streams[streamindex_receive[j]]if(x.temperatureMIN+$ $minimumdeltaTemperture) <= ascendingtemperature_GCC[i] < (x.temperatureMAX + minimumdeltaTemperture) : heatbalance_section_GCC[i]+ = -x.heatcapacity * dT$

heatbalance$_sum_GCC[i+1] = heatbalance_sum_GCC[i] - heatbalance_section_GCC[i]next$

GCCleft = np.min(heatbalance$_sum_GCC)$ カーブ左端の探索 $heatbalance_sum_GCC = heatbalance_sum_GCC-$

$GCCleftGCC$ カーブをずらす

==========

外部熱媒使用量の計算

GCCfunction = interpolate.interp1d(ascendingtemperature$_GCC, heatbalance_sum_GCC$)

def ExternalHeatFromGCCfunction(EXheatstream): 外部熱媒の使用可能量を計算する関数 EXHeat = 0.0

if EXheatstream.property == externalgive: 熱媒の場合 if EXheatstream.temperatureMIN ¿= ascendingtemperature$_GCC[-1]$: $EXHeat = heatbalance_sum_GCC[-1]else$: $EXHeat = GCCfunction(EXheatstream.temperatureMIN)foriinrange(len(ascendingtemperature_GCC))$: $ifascendingtemperature_GCC[i] >= EXheatstream.temperatureMIN : EXHeat = np.min([EXHeat, heatbalance_sum$

if EXheatstream.property == externalreceive: 冷媒の場合 if EXheatstream.temperatureMIN + minimumdeltaTemperture ¡= ascendingtemperature$_GCC[0] : EXHeat = heatbalance_sum_GCC[0]else : EXHeat = GCCfunction(EXheatstream.temperatureMIN + minimumdeltaTemperture)foriinrange(len(ascending$
$ifascendingtemperature_GCC[i] <= EXheatstream.temperatureMIN + minimumdeltaTemperture : EXHeat = np.min([EXHeat, heatbalance_sum_GCC[i]])next$

EXheatstream.deltaheat = EXHeat

for i in range (len(streamindex$_externalheat)) : 外部熱媒の使用可能量を計算 ExternalHeatFromGCCfunction(streams$

for i in range (len(streamindex$_externalgive)) : 温度の異なる外部熱媒を用いている時，熱量を分割する処理 $forjinrange(len(streamindex_externalgive)) :$ 低コスト熱媒に代用可能な部分の検索 $if(streams[streamindex_externalgive[i]].temperatureMIN > streams[streamindex_externalgive[j]].temperatureMI$
$streams[streamindex_externalgive[i]].leftend_externalheat_GCC = np.max([streams[streamindex_externalgive[i]].lefter$

for i in range (len(streamindex$_externalreceive)) : 温度の異なる外部冷媒を用いている時，熱量を分割する処理 $forjinrange(len(streamindex_externalreceive)) :$ 低コスト熱媒に代用可能な部分の検索 $ifstreams[streamindex_externalreceive[i]].temperatureMIN < streams[streamindex_externalreceive[j]].temperatur$
$streams[streamindex_externalreceive[i]] .leftend_externalheat_GCC = np.max([streams[streamindex_externalreceive[i]]$

熱容量と実際の外部熱媒使用熱量の計算 for i in range (len(streamindex$_externalheat)) : streams[streamindex_externalhea$
$-streams[streamindex_externalheat[i]].leftend_externalheat_GCCstreams[streamindex_externalheat[i]].heatcapacity_calc$

==========

TQdiagram のデータ制作==========

sectionheat$_give = np.array([0.]*(len(ascendingtemperature_TQ)))sectionheat_receive = np.array([0.]*$
$(len(ascendingtemperature_TQ)))heatsum_give = np.array([0.]*(len(ascendingtemperature_TQ)))heatsum_receive =$
$np.array([0.] * (len(ascendingtemperature_TQ)))$

for i in range (len(ascendingtemperature$_TQ) - 1) : dT = (ascendingtemperature_TQ[i +$
$1] - ascendingtemperature_TQ[i])forjinrange(len(streams)) : ifstreams[j].propertyingives :$
$ifstreams[j].temperatureindexMIN <= i < streams[j].temperatureindexMAX : sectionheat_give[i] +=$
$streams[j].heatcapacity*dTelifstreams[j].propertyinreceives : ifstreams[j].temperatureindexMIN <=$
$i < streams[j].temperatureindexMAX : sectionheat_receive[i] += streams[j].heatcapacity * dT$

next heatsum$_give[i+1] = sectionheat_give[i]+heatsum_give[i]heatsum_receive[i+1] = sectionheat_receive[i]+$
$heatsum_receive[i]next$

print("Draw TQ diagram Complete") ==========

class HeatExchanger: def $_{init_{(self):self.exchangeheat=0.0self.streamname_give=""self.streamname_receive=""self.temperaturehigh_give=0.0self.tem$

def heatcapacity$_calc(self) : self.heatcapacity_give = self.exchangeheat/(self.temperaturehigh_give -$

$self.temperaturelow_give)self.heatcapacity_receive = self.exchangeheat/(self.temperaturehigh_receive-$
$self.temperaturelow_receive)$

def deltatemperature$_LogMean_calc(self)$: $self.deltatemperature_high = np.abs(self.temperaturehigh_give-$
$self.temperaturehigh_receive)self.deltatemperature_low = np.abs(self.temperaturelow_give- self.temperaturelow_receiv$
$self.deltatemperature_low : self.LogMeantemperature = self.deltatemperature_highelse : self.LogMeantemperature =$
$(self.deltatemperature_high- self.deltatemperature_low)/ log(abs(self.deltatemperature_high/self.deltatemperature_lou$

def Rounddata(self): self.exchangeheat = np.round(self.exchangeheat,sig) self.LogMeantemperature =
np.round(self.LogMeantemperature,sig) self.heatcapacity$_give = np.round(self.heatcapacity_give)self.heatcapacity_receive$
$np.round(self.heatcapacity_receive)$

複合線図を分解して熱交換器の対応を考える TQ 関数の作成 temperature$_GCCgiveinter$ =
$interpolate.interp1d(heatsum_give,\ ascendingtemperature_TQ, fill_value =' extrapolate',\ bounds_error =$
$False)temperature_GCCreceiveinter = interpolate.interp1d(heatsum_receive,\ ascendingtemperature_TQ, fill_value ='$
$extrapolate',\ bounds_error = False)$

heatsum$_give = list(heatsum_give)heatsum_receive = list(heatsum_receive)$

def temperature$_TQgive(Q)$: 引数は $ascendingtemperature_TQ$ 参照 $ifQ[1] == give : T =$
$ascendingtemperature_TQ[Q[2]]else : T = temperature_GCCgiveinter(Q[0])return(T)$

def temperature$_TQreceive(Q) : ifQ[1] == receive : T = ascendingtemperature_TQ[Q[2]]else : T =$
$temperature_GCCreceiveinter(Q[0])return(T)$

熱量で区間分割 ascendingheat$_TQ = []foriinrange(len(ascendingtemperature_TQ)) : ascendingheat_TQ.append([heatsu$
座標, g, r どちらの区切り点か, どの昇順温度と対応しているかを保存 $ascendingheat_TQ.append([heatsum_receive[i], receive, i$
$lambdax : x[0])$

各区間での熱交換リストの作成 streamlistTQsection$_give$ = $[]streamlistTQsection_receive$ =
$[]datalist_heatexchanger = []$

countgive = -1 countreceive = -1

熱交換データの作成 for i in range (len(ascendingheat$_TQ) - 1)$:

if ascendingheat$_TQ[i][1]$ == $give$: $give, receive$ の昇順温度との対応を記録 $countgive+$ =
$1elifascendingheat_TQ[i][1] == receive : countreceive+ = 1$

temperaturelow$_give = temperature_TQgive(ascendingheat_TQ[i])temperaturelow_receive = temperature_TQreceive(asc$
$temperature_TQgive(ascendingheat_TQ[i+1])temperaturehigh_receive = temperature_TQreceive(ascendingheat_TQ[i+$
$1])deltaT_give = temperaturehigh_give - temperaturelow_givedeltaT_receive = temperaturehigh_receive -$
$temperaturelow_receive$

区間中に存在する熱流体のリストを作成 for j in range (len(streams)):

x = copy.deepcopy(streams[j])

if x.property in gives: if x.temperatureindexMIN ¡= countgive ¡ x.temperatureindexMAX:
x.deltaheat$_section = x.heatcapacity*deltaT_givestreamlistTQsection_give.append(x)ifx.propertyinreceives$:
$ifx.temperatureindexMIN <= countreceive < x.temperatureindexMAX : x.deltaheat_section =$
$x.heatcapacity * deltaT_receivestreamlistTQsection_receive.append(x)next$

与熱流体と受熱流体のマッチング while len(streamlistTQsection$_give) > 0andlen(streamlistTQsection_receive) >$
0 :

データクラス作成 dummy = HeatExchanger() datalist$_heatexchanger.append(dummy)$

heatlist$_give_receive = [streamlistTQsection_give[0].deltaheat_section,\ streamlistTQsection_receive[0].deltaheat_section]$

熱交換データの構成 dlhe = datalist$_heatexchanger[-1]dlhe.exchangeheat = np.min(heatlist_give_receive)dlhe.streamnar$

$streamlistTQsection_{g}ive[0].namedlhe.streamname_{r}eceive = streamlistTQsection_{r}eceive[0].namedlhe.temperaturehig$
$temperaturehigh_{g}ivedlhe.temperaturelow_{g}ive = temperaturelow_{g}ivedlhe.temperaturehigh_{r}eceive =$
$temperaturehigh_{r}eceivedlhe.temperaturelow_{r}eceive = temperaturelow_{r}eceivedlhe.heatcapacity_{c}alc()dlhe.phase_{g}ive =$
$streamlistTQsection_{g}ive[0].phasedlhe.phase_{r}eceive = streamlistTQsection_{r}eceive[0].phase$

if streamlistTQsection$_{g}ive[0].rebcon == "reb" or streamlistTQsection_{r}eceive[0].rebcon == "reb"$:
$dlhe.rebcon_{f}acter = 2.$

熱交換反映 streamlistTQsection$_{g}ive[0].deltaheat_{s}ection+ = -dlhe.exchangeheatstreamlistTQsection_{r}eceive[0].deltah$
$-dlhe.exchangeheat$

交換済流体の削除 deleteindex = heatlist$_{g}ive_{r}eceive.index(min(heatlist_{g}ive_{r}eceive))if deleteindex ==$
$0 : delstreamlistTQsection_{g}ive[0]else : delstreamlistTQsection_{r}eceive[0]next$

deletecounter = 0 無視小交換を取り除く処理 for i in range(len(datalist$_{h}eatexchanger)) :$
$datalist_{h}eatexchanger[i].Rounddata()if datalist_{h}eatexchanger[i].exchangeheat < eps : datalist_{h}eatexchanger.insert(0,$
$1]deletecounter+ = 1foriinrange(deletecounter) : deldatalist_{h}eatexchanger[0]$

熱交換データの整理 sorted$_{d}atalist_{h}eatexchanger = []datanumberbeforesort = len(datalist_{h}eatexchanger)whilelen(da$
$0 : mixcounter = 1sorted_{d}atalist_{h}eatexchanger.append(copy.deepcopy(datalist_{h}eatexchanger[0]))foriinrange(len(data$
1) : 交換流体と各熱容量が一致したら熱交換器をまとめる $if datalist_{h}eatexchanger[i+1].streamname_{g}ive ==$
$datalist_{h}eatexchanger[0].streamname_{g}ive \ anddatalist_{h}eatexchanger[i+1].streamname_{r}eceive ==$
$datalist_{h}eatexchanger[0].streamname_{r}eceive \ andnp.abs(datalist_{h}eatexchanger[i+1].heatcapacity_{g}ive -$
$datalist_{h}eatexchanger[0].heatcapacity_{g}ive) < eps \ andnp.abs(datalist_{h}eatexchanger[i+1].heatcapacity_{r}eceive -$
$datalist_{h}eatexchanger[0].heatcapacity_{r}eceive) < eps :$

sorted$_{d}atalist_{h}eatexchanger[-1].exchangeheat+ = datalist_{h}eatexchanger[i+1].exchangeheat$ 熱量の加算 $sorted_{d}atalist_{h}eatexchanger[-1].temperaturehigh_{g}ive = datalist_{h}eatexchanger[i+1].temperaturehigh_{g}ive$ 温度範囲の拡張 $sorted_{d}atalist_{h}eatexchanger[-1].temperaturehigh_{r}eceive =$
$datalist_{h}eatexchanger[i+1].temperaturehigh_{r}eceive$

datalist$_{h}eatexchanger.insert(0, datalist_{h}eatexchanger[i+1])deldatalist_{h}eatexchanger[i+2]mixcounter+ =$
$1nextforiinrange(mixcounter) : deldatalist_{h}eatexchanger[0]nextdatanumberaftersort = len(sorted_{d}atalist_{h}eatexchan$

totalcost=0.0 totalheat=0.0

対数平均温度差 dT, 総括伝熱係数 U, 熱交換器面積 A の計算 for i in range(len(sorted$_{d}atalist_{h}eatexchanger)) :$

sorted$_{d}atalist_{h}eatexchanger[i].deltatemperature_{L}ogMean_{c}alc()sorted_{d}atalist_{h}eatexchanger[i].Rounddata()$

for j in range (len(twophaselist)): if [sorted$_{d}atalist_{h}eatexchanger[i].phase_{g}ive, \ sorted_{d}atalist_{h}eatexchanger[i].phase_{r}e$
$twophaselist[j] \ or[sorted_{d}atalist_{h}eatexchanger[i].phase_{r}eceive, \ sorted_{d}atalist_{h}eatexchanger[i].phase_{g}ive] ==$
$twophaselist[j] : sorted_{d}atalist_{h}eatexchanger[i].U = Ulist[j]next$

sorted$_{d}atalist_{h}eatexchanger[i].area = (1.e+6/3600.)*sorted_{d}atalist_{h}eatexchanger[i].exchangeheat /(sorted_{d}atalist_{h}$
$sorted_{d}atalist_{h}eatexchanger[i].LogMeantemperature)$

sorted$_{d}atalist_{h}eatexchanger[i].cost = 0.015 * sorted_{d}atalist_{h}eatexchanger[i].rebconfacter *$
$sorted_{d}atalist_{h}eatexchanger[i].area**0.62totalcost+ = sorted_{d}atalist_{h}eatexchanger[i].costtotalheat+ =$
$sorted_{d}atalist_{h}eatexchanger[i].exchangeheatnext$

出力 wb.sheets[writesheet].clear$_{c}ontents()foriinrange(len(streams)) : wb.sheets[readsheet].range((row_{r}eadstart+$
$streams[i].number+1, \ column_{r}eadstart+2)).value = streams[i].propertywb.sheets[readsheet].range((row_{r}eadstart+$
$streams[i].number + 1, \ column_{r}eadstart + 3)).value = streams[i].deltaheat$

wb.sheets[writesheet].range((row$_{w}ritestart, column_{w}ritestart + 0)).value =$ "熱交換器番号
$"wb.sheets[writesheet].range((row_{w}ritestart, column_{w}ritestart+1)).value =$ "与熱流体"$wb.sheets[writesheet].range((r$

2)).value = ”受熱流体”wb.sheets[writesheet].range((row_writestart, column_writestart + 3)).value = ”対数平均温度差 [K]”wb.sheets[writesheet].range((row_writestart, column_writestart + 4)).value = ”交換熱量 [MJ/h]”wb.sheets[writesheet].range((row_writestart, column_writestart + 5)).value = ”伝熱面積 [m2]”wb.sheets[writesheet].range((row_writestart, column_writestart + 6)).value = ”価格 [億円]”wb.sheets[writesheet].range((row_writestart, column_writestart + 7)).value = ”総コスト [億円/year]”wb.sheets[writesheet].range((row_writestart, column_writestart + 8)).value = ”総交換熱量 [MJ/h]”wb.sheets[writesheet].range((row_writestart, column_writestart + 10)).value = ”温度差 1”wb.sheets[writesheet].range((row_writestart, column_writestart + 11)).value = ”温度差 2”

wb.sheets[writesheet].range((row_writestart+1, column_writestart+7)).value = totalcost/7.wb.sheets[writesheet].range
1, column_writestart + 8)).value = totalheat

for i in range (len(sorted_datalist_heatexchanger)) : wb.sheets[writesheet].range((row_writestart + i + 1, column_writestart + 0)).value = i + 1wb.sheets[writesheet].range((row_writestart + i + 1, column_writestart+1)).value = sorted_datalist_heatexchanger[i].streamname_givewb.sheets[writesheet].range((row_writ
i+1, column_writestart+2)).value = sorted_datalist_heatexchanger[i].streamname_receivewb.sheets[writesheet].range((ro
i+1, column_writestart+3)).value = sorted_datalist_heatexchanger[i].LogMeantemperaturewb.sheets[writesheet].range(
i+1, column_writestart+4)).value = sorted_datalist_heatexchanger[i].exchangeheatwb.sheets[writesheet].range((row_writ
i+1, column_writestart+5)).value = sorted_datalist_heatexchanger[i].areawb.sheets[writesheet].range((row_writestart+
i + 1, column_writestart + 6)).value = sorted_datalist_heatexchanger[i].cost

wb.sheets[writesheet].range((row_writestart+i+1, column_writestart+10)).value = sorted_datalist_heatexchanger[i].delta
i + 1, column_writestart + 11)).value = sorted_datalist_heatexchanger[i].deltatemperature_lownext

==========

グランドコンポジットカーブのデータ出力

rcParams['font.family'] = 'sans-serif' rcParams['font.sans-serif'] = ['Hiragino Maru Gothic Pro', 'Yu Gothic', 'Meirio', 'Takao', 'IPAexGothic', 'IPAPGothic', 'VL PGothic', 'Noto Sans CJK JP'] rcParams['font.size'] = 22

fig1=plt.figure() ax1 = fig1.add_subplot()ax1.plot(heatbalance_sum_GCC, ascendingtemperature_GCC, label = ”グランドコンポジットカーブ”)plt.xlim(0, 40)plt.ylim(0, 250)ax1.set_xlabel(xlabel = ”熱量 [MJ/h]”)ax1.set_ylabel(ylabel = ”温度 [℃]”)plt.gca().xaxis.set_tick_params(which =′ both′, direction =′ in′, bottom = True, top = True, left = True, right = True)plt.gca().yaxis.set_tick_params(which =′ both′, direction =′ in′, bottom = True, top = True, left = True, right = True)plt.rcParams[”legend.fancybox”] = False 丸角 plt.rcParams[”legend.framealpha”] = 1 透明度の指定, 0 で塗りつぶしなし plt.rcParams[”legend.edgecolor”] = black′edge の色を変更 plt.legend()plt.show()

print(”Draw GCC diagram Complete”)

TQ 線図出力 fig2 = plt.figure() ax_give = fig2.add_subplot()ax_receive = fig2.add_subplot()heatsum_give = list(heatsum_give)heatsum_receive = list(heatsum_receive)ascendingtemperature_TQ = list(ascendingtemperature_TQ)

ax_give.plot(heatsum_give, ascendingtemperature_TQ, color = ”r”)ax_receive.plot(heatsum_receive, ascendingtemperatu ”b”)plt.xlim(0, heatsum_receive[−1])plt.ylim(−10, ascendingtemperature_TQ[−1]+10)ax_give.set_xlabel(xlabel = ”熱量 [MJ/h]”)ax_give.set_ylabel(ylabel = ”温度 [℃]”)plt.gca().xaxis.set_tick_params(which =′ both′, direction =′ in′, bottom = True, top = True, left = True, right = True)plt.gca().yaxis.set_tick_params(which =′ both′, direction =′ in′, bottom = True, top = True, left = True, right = True)plt.rcParams[”legend.fancybox”] = False 丸角 plt.rcParams[”legend.framealpha”] = 1 透明度の指定, 0 で塗りつぶしなし plt.rcParams[”legend.edgecolor”] = black′edge の色を変更 plt.legend()plt.tick_params(axis =′ x′, colors =′ k′)plt.tick_params(axis =′

$y', colors =' k')plt.show()$

B.2 VBA

[caption=変数宣言] Option Explicit

Public Const TolMolarMass = 92.141 ' トルエンモル質量 [kg/kmol] Public Const BzAMolarMass = 122.124 ' 安息香酸モル質量 [kg/kmol] Public Const WaterMolarMass = 18.015 ' 水モル質量 [kg/kmol]

Public Const OxygenFraction = 0.2 Public Const NytrogenFraction = 0.8

Public hycase As Object

Public AdditionalAirHYSYS As ProcessStream Public AdopterHYSYS As ProcessStream Public BoilerBottomOutHYSYS As ProcessStream Public BoilerTopOutHYSYS As ProcessStream Public CrystallizationInHYSYS As ProcessStream Public CrystallizedRecycleHYSYS As ProcessStream Public DecanterOutHYSYS As ProcessStream Public DistillationInHYSYS As ProcessStream Public DistillationLiqOutHYSYS As ProcessStream Public DistillationVapOutHYSYS As ProcessStream Public DistVapOutXHYSYS As ProcessStream Public ExternalOutHYSYS As ProcessStream Public ExtractedBzAHYSYS As ProcessStream Public ExtractedOthersHYSYS As ProcessStream Public ExtractedWaterHYSYS As ProcessStream Public FeedHYSYS As ProcessStream Public MixerInHYSYS As ProcessStream Public PostDistillationVapOutHYSYS As ProcessStream Public PostDistVapOutX-HYSYS As ProcessStream Public PostReactorOutHYSYS As ProcessStream Public PreAdopterHYSYS As ProcessStream Public PreCrystallizedRecycleHYSYS As ProcessStream Public PreReactorInHYSYS As ProcessStream Public ProductHYSYS As ProcessStream Public PumpInHYSYS As ProcessStream Public ReactorInHYSYS As ProcessStream Public ReactorOutHYSYS As ProcessStream Public ValveOutHYSYS As ProcessStream

Public PreReactorInHeaterHYSYS As ProcessStream Public ReactorOutPumpHYSYS As ProcessStream Public CondenserHYSYS As ProcessStream Public ReboilerHYSYS As ProcessStream Public ExternalOutCoolerHYSYS As ProcessStream Public PreCrystallizedRecycleHeaterHYSYS As ProcessStream Public ValveOutCoolerHYSYS As ProcessStream Public PowerOfPumpHYSYS As ProcessStream Public DistVapOutCoolerHYSYS As ProcessStream Public DistVapOutExpandHYSYS As ProcessStream

[caption=HYSYS と python を繋ぐコード] Option Explicit

Sub main()

Dim topRow As Integer topRow = 18

Dim resultSheet As Worksheet Set resultSheet = ThisWorkbook.Sheets(1)

With resultSheet .Cells(10, 4) = "" .Cells(11, 4) = "" .Range(.Cells(18, 4), .Cells(48, 12)) = "" .Range(.Cells(18, 11), .Cells(50, 12)) = 0 End With

Dim i As Integer Dim j As Integer Dim m As Integer Dim n As Integer Dim h As Integer Dim p As Integer Dim q As Integer

Dim flowRange As Variant 'excel シートから array への adopter Dim reactorErr As Double ' 反応器収束用誤差 Dim reactorEps As Double ' 反応器収束用 Dim crystallizerErr As Double ' 晶析器収束用誤差 Dim crystallizerEps As Double ' 晶析器収束用

reactorErr = 1 crystallizerErr = 1 reactorEps = 0.001 crystallizerEps = 0.001

'———————————————————— Dim k As Variant '[h-1] 反応速度定数 Dim

reactorPres As Double '[bar] 反応器圧力 Dim reactorVol As Double '[m3] 反応器体積 Dim reactorTemp As Double '[℃] 反応器温度 Dim F0 As Double '[kmol/h] feed トルエン流量 Dim extractorTemp As Double '[℃] 抽出温度 Dim crystallizerVolume As Double '[℃] 晶析器体積 Dim crystallizerTemp As Double '[℃] 晶析器温度

With resultSheet reactorPres = .Cells(3, 4) reactorVol = .Cells(4, 4) reactorTemp = .Cells(5, 4) + 273.15 F0 = .Cells(6, 4) extractorTemp = 40 crystallizerVolume = .Cells(8, 4) crystallizerTemp = .Cells(9, 4) End With Call Utils.solve$_k(reactorTemp, k)' --$

'———————————————————————— Set hycase = GetObject(ThisWorkbook.Path "¥ main.hsc")

With hycase.Flowsheet.MaterialStreams Set AdditionalAirHYSYS = .Item("AdditionalAir") Set AdopterHYSYS = .Item("Adopter") Set BoilerBottomOutHYSYS = .Item("BoilerBottomOut") Set BoilerTopOutHYSYS = .Item("BoilerTopOut") Set CrystallizationInHYSYS = .Item("CrystallizationIn") Set CrystallizedRecycleHYSYS = .Item("CrystallizedRecycle") Set DecanterOutHYSYS = .Item("DecanterOut") Set DistillationInHYSYS = .Item("DistillationIn") Set DistillationLiqOutHYSYS = .Item("DistillationLiqOut") Set DistillationVapOutHYSYS = .Item("DistillationVapOut") Set DistVapOutXHYSYS = .Item("DistVapOutX") Set ExternalOutHYSYS = .Item("ExternalOut") Set ExtractedBzAHYSYS = .Item("ExtractedBzA") Set ExtractedOthersHYSYS = .Item("ExtractedOthers") Set ExtractedWaterHYSYS = .Item("ExtractedWater") Set FeedHYSYS = .Item("Feed") Set MixerInHYSYS = .Item("MixerIn") Set PostDistillationVapOutHYSYS = .Item("PostDistillationVapOut") Set PostDistVapOutXHYSYS = .Item("PostDistVapOutX") Set PostReactorOutHYSYS = .Item("PostReactorOut") Set PreAdopterHYSYS = .Item("PreAdopter") Set PreCrystallizedRecycleHYSYS = .Item("PreCrystallizedRecycle") Set PreReactorInHYSYS = .Item("PreReactorIn") Set ProductHYSYS = .Item("Product") Set PumpInHYSYS = .Item("PumpIn") Set ReactorInHYSYS = .Item("ReactorIn") Set ReactorOutHYSYS = .Item("ReactorOut") Set ValveOutHYSYS = .Item("ValveOut") End With

With hycase.Flowsheet.EnergyStreams Set PreReactorInHeaterHYSYS = .Item("PreReactorInHeater") Set ReactorOutPumpHYSYS = .Item("ReactorOutPump") Set CondenserHYSYS = .Item("Condenser") Set ReboilerHYSYS = .Item("Reboiler") Set ExternalOutCoolerHYSYS = .Item("ExternalOutCooler") Set PreCrystallizedRecycleHeaterHYSYS = $.Item("PreCrystallizedRecycleHeater")SetValveOutCoolerHYSYS = .Item("ValveOutCooler")SetPowerOfPumpHYSYS = .Item("PowerOfPump")SetDistVapOutCoolerHYSYS = .Item("DistVapOutCooler")SetDistVapOutExpandHYSYS = .Item("DistVapOutExpand")EndWith$

Dim DistillationHYSYS As ColumnOp

Dim ToReb As ProcessStream Dim BoilUp As ProcessStream Dim ToCond As ProcessStream Dim Reflux As ProcessStream

Set DistillationHYSYS = hycase.Flowsheet.Operations.Item("T-100") With DistillationHYSYS.ColumnFlowsheet.Mate Set ToReb = .Item("To Reboiler") Set BoilUp = .Item("Boilup") Set ToCond = .Item("To Condenser") Set Reflux = .Item("Reflux") End With '————————————————————————

'———————————————————————— With resultSheet AdditionalAirHYSYS.Pressure.SetValue .Cells(topRow + 0, 1), "bar" AdopterHYSYS.Pressure.SetValue .Cells(topRow + 1, 1), "bar" CrystallizedRecycleHYSYS.Pressure.SetValue .Cells(topRow + 5, 1), "bar" DecanterOutHYSYS.Pressure.SetValue .Cells(topRow + 6, 1), "bar" DistillationInHYSYS.Pressure.SetValue .Cells(topRow + 7, 1), "bar" DistVapOutXHYSYS.Pressure.SetValue .Cells(topRow + 10, 1), "bar" ExternalOutHYSYS.Pressure.SetValue

.Cells(topRow + 11, 1), "bar" ExtractedBzAHYSYS.Pressure.SetValue .Cells(topRow + 12, 1), "bar" ExtractedOthersHYSYS.Pressure.SetValue .Cells(topRow + 13, 1), "bar" ExtractedWaterHYSYS.Pressure.SetValue .Cells(topRow + 14, 1), "bar" FeedHYSYS.Pressure.SetValue .Cells(topRow + 15, 1), "bar" PostDistillationVapOutHYSYS.Pressure.SetValue .Cells(topRow + 17, 1), "bar" PostDistVapOutXHYSYS.Pressure.SetValue .Cells(topRow + 18, 1), "bar" PostReactorOutHYSYS.Pressure.SetValue .Cells(topRow + 19, 1), "bar" PreAdopterHYSYS.Pressure.SetValue .Cells(topRow + 20, 1), "bar" PreCrystallizedRecycleHYSYS.Pressure.SetValue .Cells(topRow + 21, 1), "bar" PreReactorInHYSYS.Pressure.SetValue .Cells(topRow + 22, 1), "bar" ProductHYSYS.Pressure.SetValue .Cells(topRow + 23, 1), "bar" ReactorInHYSYS.Pressure.SetValue .Cells(topRow + 25, 1), "bar" ReactorOutHYSYS.Pressure.SetValue .Cells(topRow + 26, 1), "bar" ValveOutHYSYS.Pressure.SetValue .Cells(topRow + 27, 1), "bar"

AdditionalAirHYSYS.Temperature.SetValue .Cells(topRow + 0, 2), "C" AdopterHYSYS.Temperature.SetValue .Cells(topRow + 1, 2), "C" CrystallizedRecycleHYSYS.Temperature.SetValue .Cells(topRow + 5, 2), "C" DecanterOutHYSYS.Temperature.SetValue .Cells(topRow + 6, 2), "C" ExternalOutHYSYS.Temperature.SetValue .Cells(topRow + 11, 2), "C" ExtractedBzAHYSYS.Temperature.SetValue .Cells(topRow + 12, 2), "C" ExtractedOthersHYSYS.Temperature.SetValue .Cells(topRow + 13, 2), "C" ExtractedWaterHYSYS.Temperature.SetValue .Cells(topRow + 14, 2), "C" FeedHYSYS.Temperature.SetValue .Cells(topRow + 15, 2), "C" PreAdopterHYSYS.Temperature.SetValue .Cells(topRow + 20, 2), "C" PreCrystallizedRecycleHYSYS.Temperature.SetValue .Cells(topRow + 21, 2), "C" ProductHYSYS.Temperature.SetValue .Cells(topRow + 23, 2), "C" ReactorInHYSYS.Temperature.SetValue .Cells(topRow + 25, 2), "C" ReactorOutHYSYS.Temperature.SetValue .Cells(topRow + 26, 2), "C" End With '——

'———————————————————————————————— Dim feedMolarFlow As Double 'feed モル流量 [kmol/h] Dim feedVolumeFlow As Double 'feed 体積流量 [m3/h] Dim feedComponentMolarFlow As Variant 'feed 成分別モル流量 [kmol/h] Dim feedComponentVolumeFlow As Variant 'feed 成分別体積流量 [m3/h] Dim feedComponentMolarFractin As Variant 'feed モル分率 [-]

Dim reactorInMolarFlow As Double 'CSTR 入口モル流量 [kmol/h] Dim reactorInVolumeFlow As Double 'CSTR 入口体積流量 [m3/h] Dim reactorInComponentMolarFlow As Variant 'CSTR 入口成分別モル流量 [kmol/h] Dim reactorInComponentVolumeFlow As Variant 'CSTR 入口成分別体積流量 [m3/h] Dim reactorInComponentMolarFraction As Variant 'CSTR 入口モル分率 [-]

Dim reactorOutMolarFlow As Double 'CSTR 出口モル流量 [kmol/h] Dim reactorOutVolumeFlow As Double 'CSTR 出口体積流量 [m3/h] Dim reactorOutComponentMolarFlow As Variant 'CSTR 出口成分別モル流量 [kmol/h] Dim reactorOutComponentVolumeFlow As Variant 'CSTR 出口成分別体積流量 [m3/h] Dim reactorOutComponentMolarFraction As Variant 'CSTR 出口モル分率 [-]

Dim decanterOutComponentMolarFlow As Variant ' デカンタ気体出口成分別モル流量 [kmol/h]

Dim boilerInComponentMolarFlow As Variant ' ボイラー入口成分別モル流量 [kmol/h] Dim distillationLiqOutComponentMolarFlow As Variant ' 蒸留塔缶出液成分別モル流量 [kmol/h]

Dim extractedWaterMolarFlow As Double ' 抽出器出口水モル流量 Dim extractedWaterComponentMolarFlow As Variant ' 抽出器出口水成分別モル流量

Dim extractedBzAMolarFlow As Double ' 抽出器出口安息香酸モル流量 Dim extractedBzAComponentMolarFlow As Variant ' 抽出器出口安息香酸成分別モル流量

Dim extractedOthersMolarFlow As Double ' 抽出器出口不純物モル流量 Dim extractedOthersCompo-

nentMolarFlow As Variant ’ 抽出器出口不純物成分別モル流量

Dim adopterComponentMolarFlow As Variant ’ リサイクル計算用蒸留塔缶出液流量 [kmol/h]

Dim ToCrystallizationMolarFlow As Double ’ 晶析器入口モル流量 [kmol/h] Dim ToCrystallizationVolumeFlow As Double ’ 晶析器入口体積流量 [m3/h] Dim ToCrystallizationBzAMolarFraction As Double ’ 晶析器入口安息香酸モル分率 [-] Dim ToCrystallizationBzAMolarFlow As Double ’ 晶析器入口安息香酸モル流量 [kmol/h] Dim ToCrystallizationWaterMolarFlow As Double ’ 晶析器入口水モル流量 [kmol/h]

Dim BzAConcentration As Double ’ 晶析器入口安息香酸濃度 [kg/m3] Dim CrystallizedBzA As Double ’ 晶析した安息香酸モル流量 [kmole/h] Dim products As Variant ’ 晶析した安息香酸 Dim restOfProducts As Variant ’ 晶析しなかった残り Dim RestOfBzA As Double ’ 晶析しなかった安息香酸 Dim RestOfWater As Double ’ 晶析器から流出する水

Dim necessaryOxygen As Double ’ トルエン完全燃焼に必要な酸素 [kmol/h] Dim lackingOxygen As Double ’ 不足している酸素 [kmol/h] Dim additionalOxygen As Double ’ 追加する酸素 [kmol/h] Dim additionalNytrogen As Double ’ 追加する窒素 [kmol/h] Dim additionalAir As Variant ’ 追加する空気 [kmol/h] Dim isAirNeed As Boolean ’whether additional air is needed isAirNeed = False

Dim distillationVapOutTmp As Double ’ 蒸留塔留出液の温度 Dim distillationVapOutComponentMolarFlow As Variant ’ 蒸留塔留出液の成分別モル流量 [kmol/h] ’————————————————————————

’———————————————————— feedComponentMolarFlow = Array(F0, 0, 0, 0, 0, 0, 0, 0, 0) feedComponentVolumeFlow = Array(0, 0, 0, 0, 0, 0, 0, 0, 0) feedComponentMolarFractin = Array(1, 0, 0, 0, 0, 0, 0, 0, 0)

reactorInComponentMolarFlow = Array(F0, 0, 0, 0, 0, 0, 0, 0, 0) reactorInComponentVolumeFlow = Array(0, 0, 0, 0, 0, 0, 0, 0, 0) reactorInComponentMolarFraction = Array(1, 0, 0, 0, 0, 0, 0, 0, 0) Call Utils.solveComponentVolumeFlow(reactorInComponentMolarFlow, $_r eactorInComponentVolumeFlow)reactorInMolarFlow = Utils.sumOfFlow(reactorInComponentMolarFlow)react$
$Utils.sumOfFlow(reactorInComponentVolumeFlow)$

reactorOutComponentMolarFlow = Array(0, 0, 0, 0, 0, 0, 0, 0, 0) reactorOutComponentVolumeFlow = Array(0, 0, 0, 0, 0, 0, 0, 0, 0) reactorOutComponentMolarFraction = Array(0, 0, 0, 0, 0, 0, 0, 0, 0)

boilerInComponentMolarFlow = Array(0, 0, 0, 0, 0, 0, 0, 0, 0) decanterOutComponentMolarFlow = Array(0, 0, 0, 0, 0, 0, 0, 0, 0)

distillationLiqOutComponentMolarFlow = Array(0, 0, 0, 0, 0, 0, 0, 0, 0)

extractedBzAComponentMolarFlow = Array(0, 0, 0, 0, 0, 0, 0, 0, 0) extractedOthersComponentMolarFlow = Array(0, 0, 0, 0, 0, 0, 0, 0, 0) extractedWaterComponentMolarFlow = Array(0, 0, 0, 0, 0, 0, 0, 0, 0)

adopterComponentMolarFlow = Array(0, 0, 0, 0, 0, 0, 0, 0, 0)

distillationVapOutComponentMolarFlow = Array(0, 0, 0, 0, 0, 0, 0, 0, 0) ’————————————————————————

’———————————————————— FeedHYSYS.ComponentMolarFlow.SetValues feedComponentMolarFlow, ”kgmole/h” i = 0 Do While (reactorErr ¿ reactorEps And i ¡ 600) With resultSheet .Range(.Cells(topRow + 25, 4), .Cells(topRow + 25, 12)) = $_r eactorInComponentMolarFlow.Cells(topRow+$
$25, 14) = reactorInVolumeFlow'単通反応率.Cells(12, 4) = (reactorInComponentMolarFlow(0) -_r$
$eactorOutComponentMolarFlow(0))/_r eactorInComponentMolarFlow(0)EndWith$

Call RunPython(”import analyzer$_n owrite; analyzer_n owrite.main()$”)

' 反応器出口のモル流量をシートから取得する．resultSheet.Cells(topRow + 26, 9) = 0 resultSheet.Cells(topRow + 26, 10) = 0 flowRange = resultSheet.Range(resultSheet.Cells(topRow + 26, 4), $_{r}esultSheet.Cells(topRow + 26, 12))Forj = 1To9reactorOutComponentMolarFlow(j - 1) = flowRange(1, j)Nextjreactor OutComponentMolarFlow(5) = 0reactorOutComponentMolarFlow(6) = 0$

' デカンタから蒸発するモル流量をシートから取得する．flowRange = resultSheet.Range(resultSheet.Cells(topRow + 31, 4), $_{r}esultSheet.Cells(topRow + 31, 12))Forj = 1To9decanterOutComponentMolarFlow(j - 1) = flowRange(1, j)Nextj$

' HYSYS に反応器出口とデカンタ出口のモル流量を渡す．ReactorOutHYSYS.ComponentMolarFlow.SetValues $_{r}eactorOutComponentMolarFlow,"kgmole/h"DecanterOutHYSYS.ComponentMolarFlow.SetValues_{d}ecanterOutC$

' 蒸留塔留出液のつじつま合わせ．distillationVapOutTmp = PostDistillationVapOutHYSYS. $_{T}emperature.GetValue("C")distillationVapOutComponentMolarFlow = PostDistillationVapOutHYSYS._{C}$ompone 0$distillationVapOutComponentMolarFlow(6) = 0$

DistVapOutXHYSYS.Temperature.SetValue distillationVapOutTmp, "C" DistVapOutXHYSYS.ComponentMolarFlow $_{d}istillationVapOutComponentMolarFlow,"kgmole/h"$

' 蒸留塔缶出液のモル流量を HYSYS から取得し，疑似抽出を行う．distillationLiqOutComponentMolarFlow = DistillationLiqOutHYSYS. $_{C}omponentMolarFlow.GetValues("kgmole/h")$

Call pseudo$_{e}xtractor.main(distillationLiqOutComponentMolarFlow,_{e}xtractedBzAComponentMolarFlow,_{e}xtracte$

ExtractedBzAHYSYS.ComponentMolarFlow.SetValues $_{e}xtractedBzAComponentMolarFlow,"kgmole/h"ExtractedC$

' 疑似抽出の結果を晶析器に渡す．ToCrystallizationBzAMolarFlow = extractedBzAComponentMolarFlow(3) ToCrystallizationWaterMolarFlow = extractedWaterComponentMolarFlow(4) ToCrystallizationVolumeFlow = CrystallizationInHYSYS. $_{S}tdLiqVolFlow.GetValue("m3/h")$

BzAConcentration = (ToCrystallizationBzAMolarFlow * BzAMolarMass) / $_{(}ToCrystallizationWaterMolarFlow* WaterMolarMass)$

CrystallizedBzA = crystallization.MSMPR(crystallizerTemp, $_{B}zAConcentration,_{T}oCrystallizationVolumeFlow,_{c}rys$

RestOfBzA = ToCrystallizationBzAMolarFlow - CrystallizedBzA RestOfWater = ToCrystallizationWaterMolarFlow products = Array(0, 0, 0, CrystallizedBzA, 0, 0, 0, 0, 0) restOfProducts = Array(0, 0, 0, RestOfBzA, RestOfWater, 0, 0, 0, 0) ProductHYSYS.ComponentMolarFlow.SetValues products, "kgmole/h" PreCrystallizedRecycleHYSYS.ComponentMolarFlow.SetValues $_{r}estOfProducts,"kgmole/h"$

' 収束判定を行う．reactorErr = Abs((distillationLiqOutComponentMolarFlow(3) + $_{d}ecanterOutComponentMolarFlou F0))/F0$

reactorInComponentMolarFlow = ReactorInHYSYS. $_{C}omponentMolarFlow.GetValues("kgmole/h")reactorInVolum ReactorInHYSYS.StdLiqVolFlow.GetValue("m3/h")$

i = i + 1 Loop

'———-反応器収束後，シートに計算結果を書き込むバージョンの python を実行する．——— Call RunPython("import analyzer$_{f}inal;analyzer_{f}inal.main()")resultSheet.Cells(topRow + 26, 9) = 0resultSheet.Cells(topRow + 26, 10) = 0flowRange = resultSheet.Range(resultSheet.Cells(topRow + 26, 4),_{r}esultSheet.Cells(topRow + 26, 12))Forj = 1To9reactorOutComponentMolarFlow(j - 1) = flowRange(1, j)Nextjflow Range = resultSheet.Range(resultSheet.Cells(topRow+31, 4),_{r}esultSheet.Cells(topRow+ 31, 12))Forj = 1To9decanterOutComponentMolarFlow(j-1) = flowRange(1, j)NextjreactorOutComponentMolarF 0ReactorOutHYSYS.ComponentMolarFlow.SetValues_{r}eactorOutComponentMolarFlow,"kgmole/h"DecanterOutH$

distillationLiqOutComponentMolarFlow = $_{D}istillationLiqOutHYSYS.ComponentMolarFlow.GetValues("kgmole/h$

Call pseudo$_extractor.main(distillationLiqOutComponentMolarFlow,_extractedBzAComponentMolarFlow,_extracte$

ExtractedWaterHYSYS.ComponentMolarFlow.SetValues $_extractedWaterComponentMolarFlow,"kgmole/h" Extract$

ToCrystallizationBzAMolarFlow = extractedBzAComponentMolarFlow(3) ToCrystallizationWaterMolarFlow = extractedWaterComponentMolarFlow(4) ToCrystallizationVolumeFlow = CrystallizationInHYSYS. $_StdLiqVolFlow.GetValue("m3/h")$

BzAConcentration = (ToCrystallizationBzAMolarFlow * BzAMolarMass) / $_(ToCrystallizationWaterMolarFlow* WaterMolarMass)$

CrystallizedBzA = crystallization.MSMPR(crystallizerTemp, $_BzAConcentration,_ToCrystallizationVolumeFlow,_crys$

RestOfBzA = ToCrystallizationBzAMolarFlow - CrystallizedBzA RestOfWater = ToCrystallizationWaterMolarFlow products = Array(0, 0, 0, CrystallizedBzA, 0, 0, 0, 0, 0) restOfProducts = Array(0, 0, 0, RestOfBzA, RestOfWater, 0, 0, 0, 0) ProductHYSYS.ComponentMolarFlow.SetValues products, "kgmole/h" PreCrystallizedRecycleHYSYS.ComponentMolarFlow.SetValues $_restOfProducts,"kgmole/h"$

'————————————————————————

'——————抽出器の収束計算を行う—————— ' 蒸留塔出口のモル流量は操作できないので，晶析のリサイクルを行うために，' Adopter を設置し，そこにで流量を操作する．' まず，preAdopter の成分別モル流量を設定する．AdopterHYSYS.ComponentMolarFlow.SetValues $_PreAdopterHYSYS.ComponentMolarFlow.GetValues("kgmole/h"),"kgmole/h"$

i = 0 Do While (crystallizerErr ¿ crystallizerEps And i ¡ 100) adopterComponentMolarFlow = AdopterHYSYS. $_ComponentMolarFlow.GetValues("kgmole/h")$

Call pseudo$_extractor.main(adopterComponentMolarFlow,_extractedBzAComponentMolarFlow,_extractedOthersCo$

ExtractedWaterHYSYS.ComponentMolarFlow.SetValues $_extractedWaterComponentMolarFlow,"kgmole/h" Extract$

ToCrystallizationBzAMolarFlow = extractedBzAComponentMolarFlow(3) ToCrystallizationWaterMolarFlow = extractedWaterComponentMolarFlow(4) ToCrystallizationVolumeFlow = CrystallizationInHYSYS. $_StdLiqVolFlow.GetValue("m3/h")$

BzAConcentration = (ToCrystallizationBzAMolarFlow * BzAMolarMass) / $_(ToCrystallizationWaterMolarFlow* WaterMolarMass)$

CrystallizedBzA = crystallization.MSMPR(crystallizerTemp, $_BzAConcentration,_ToCrystallizationVolumeFlow,_crys$

RestOfBzA = ToCrystallizationBzAMolarFlow - CrystallizedBzA RestOfWater = ToCrystallizationWaterMolarFlow products = Array(0, 0, 0, CrystallizedBzA, 0, 0, 0, 0, 0) restOfProducts = Array(0, 0, 0, RestOfBzA, RestOfWater, 0, 0, 0, 0) ProductHYSYS.ComponentMolarFlow.SetValues products, "kgmole/h" PreCrystallizedRecycleHYSYS.ComponentMolarFlow.SetValues $_restOfProducts,"kgmole/h"$

adopterComponentMolarFlow(3) = distillationLiqOutComponentMolarFlow(3) + $_restOfProducts(3)AdopterHYSYS.$

crystallizerErr = Abs((distillationLiqOutComponentMolarFlow(3) + $_restOfProducts(3)) - (products(3)+restOfProducts(3)))i = i+1Loop' -$

'———————————————————————— decanterOutComponentMolarFlow = $_DecanterOutHYSYS.ComponentMolarFlow.GetValues("kgmole/h")$

' 量論関係から，トルエンを全量燃焼させるのに必要な酸素量を求める．' Tol + 9 O2 -¿ 4 H2O + 7 CO2 necessaryOxygen = decanterOutComponentMolarFlow(0) * 9 lackingOxygen = necessaryOxygen - decanterOutComponentMolarFlow(6) additionalOxygen = lackingOxygen additionalNytrogen = lackingOxygen * (NytrogenFraction / OxygenFraction)

If (additionalOxygen ¡ 0) Then additionalOxygen = 0 additionalNytrogen = 0 isAirNeed = True End

If

additionalAir = Array(0, 0, 0, 0, 0, additionalNytrogen, additionalOxygen, 0, 0) AdditionalAirHYSYS.ComponentMolarFlow.SetValues additionalAir, "kgmole/h" '————————————————

'———————————————————— With resultSheet For i = 0 To 8 .Cells(topRow + 0, i + 4) = $_A$additionalAirHYSYS.ComponentMolarFlow(i) * 3600.Cells(topRow + 1, i+4) $=_A$ dopterHYSYS.ComponentMolarFlow(i)*3600.Cells(topRow+2, i+4) $=_B$ oilerBottomOutHYSYS.Compon
3600.Cells(topRow+3, i+4) $=_B$ oilerTopOutHYSYS.ComponentMolarFlow(i)*3600.Cells(topRow+ 4, i + 4) $=_C$ rystallizationInHYSYS.ComponentMolarFlow(i) * 3600.Cells(topRow + 5, i + 4) $=_C$ rystallizedRecycleHYSYS.ComponentMolarFlow(i) * 3600.Cells(topRow + 6, i + 4) $=_D$ ecanterOutHYSYS.ComponentMolarFlow(i)*3600.Cells(topRow+7, i+4) $=_D$ istillationInHYSYS.ComponentMola
3600.Cells(topRow+8, i+4) $=_D$ istillationLiqOutHYSYS.ComponentMolarFlow(i)*3600.Cells(topRow+ 9, i + 4) $=_D$ istillationVapOutHYSYS.ComponentMolarFlow(i) * 3600.Cells(topRow + 10, i + 4) $=_D$ istVapOutXHYSYS.ComponentMolarFlow(i) * 3600.Cells(topRow + 11, i + 4) $=_E$ xternalOutHYSYS.ComponentMolarFlow(i)*3600.Cells(topRow+12, i+4) $=_E$ xtractedBzAHYSYS.ComponentMol
3600.Cells(topRow+13, i+4) $=_E$ xtractedOthersHYSYS.ComponentMolarFlow(i)*3600.Cells(topRow+ 14, i + 4) $=_E$ xtractedWaterHYSYS.ComponentMolarFlow(i) * 3600.Cells(topRow + 15, i + 4) $=_F$ eedHYSYS.ComponentMolarFlow(i)*3600.Cells(topRow+16, i+4) $=_M$ ixerInHYSYS.ComponentMolarFlow(i)* 3600.Cells(topRow + 17, i + 4) $=_P$ ostDistillationVapOutHYSYS.ComponentMolarFlow(i) * 3600.Cells(topRow+18, i+4) $=_P$ ostDistVapOutXHYSYS.ComponentMolarFlow(i)*3600.Cells(topRow+ 19, i + 4) $=_P$ ostReactorOutHYSYS.ComponentMolarFlow(i) * 3600.Cells(topRow + 20, i + 4) $=_P$ reAdopterHYSYS.ComponentMolarFlow(i)*3600.Cells(topRow+21, i+4) $=_P$ reCrystallizedRecycleHYSYS.Compon
3600.Cells(topRow+22, i+4) $=_P$ reReactorInHYSYS.ComponentMolarFlow(i)*3600.Cells(topRow+ 23, i + 4) $=_P$ roductHYSYS.ComponentMolarFlow(i) * 3600.Cells(topRow + 24, i + 4) $=_P$ umpInHYSYS.ComponentMolarFlow(i)*3600.Cells(topRow+25, i+4) $=_R$ eactorInHYSYS.ComponentMolarFlow(
3600.Cells(topRow + 26, i + 4) $=_R$ eactorOutHYSYS.ComponentMolarFlow(i) * 3600.Cells(topRow + 27, i + 4) $=_V$ alveOutHYSYS.ComponentMolarFlow(i) * 3600Next

.Cells(topRow + 0, 13) = $_A$additionalAirHYSYS.molarFlow.GetValue("kgmole/h").Cells(topRow+ 1, 13) $=_A$ dopterHYSYS.molarFlow.GetValue("kgmole/h").Cells(topRow+2, 13) $=_B$ oilerBottomOutHYSYS.molarF
3, 13) $=_B$ oilerTopOutHYSYS.molarFlow.GetValue("kgmole/h").Cells(topRow + 4, 13) $=_C$ rystallizationInHYSYS.molarFlow.GetValue("kgmole/h").Cells(topRow+5, 13) $=_C$ rystallizedRecycleHYSYS.mola
6, 13) $=_D$ ecanterOutHYSYS.molarFlow.GetValue("kgmole/h").Cells(topRow + 7, 13) $=_D$ istillationInHYSYS.molarFlow.GetValue("kgmole/h").Cells(topRow+8, 13) $=_D$ istillationLiqOutHYSYS.molarFlo
9, 13) $=_D$ istillationVapOutHYSYS.molarFlow.GetValue("kgmole/h").Cells(topRow + 10, 13) $=_D$ istVapOutXHYSYS.molarFlow.GetValue("kgmole/h").Cells(topRow+11, 13) $=_E$ xternalOutHYSYS.molarFlow.G
12, 13) $=_E$ xtractedBzAHYSYS.molarFlow.GetValue("kgmole/h").Cells(topRow + 13, 13) $=_E$ xtractedOthersHYSYS.molarFlow.GetValue("kgmole/h").Cells(topRow+14, 13) $=_E$ xtractedWaterHYSYS.molarF
15, 13) $=_F$ eedHYSYS.molarFlow.GetValue("kgmole/h").Cells(topRow+16, 13) $=_M$ ixerInHYSYS.molarFlow.GetV
17, 13) $=_P$ ostDistillationVapOutHYSYS.molarFlow.GetValue("kgmole/h").Cells(topRow + 18, 13) $=_P$ ostDistVapOutXHYSYS.molarFlow.GetValue("kgmole/h").Cells(topRow + 19, 13) $=_P$ ostReactorOutHYSYS.molarFlow.GetValue("kgmole/h").Cells(topRow+20, 13) $=_P$ reAdopterHYSYS.molarFlow.G
21, 13) $=_P$ reCrystallizedRecycleHYSYS.molarFlow.GetValue("kgmole/h").Cells(topRow +

22, 13) =P reReactorInHYSYS.molarFlow.GetValue("kgmole/h").Cells(topRow + 23, 13) =P
roductHYSYS.molarFlow.GetValue("kgmole/h").Cells(topRow+24, 13) =P umpInHYSYS.molarFlow.GetValue("k
25, 13) =R eactorInHYSYS.molarFlow.GetValue("kgmole/h").Cells(topRow + 26, 13) =R
eactorOutHYSYS.molarFlow.GetValue("kgmole/h").Cells(topRow+27, 13) =V alveOutHYSYS.molarFlow.GetValu
.Cells(topRow + 0, 14) = AdditionalAirHYSYS.StdLiqVolFlow.GetValue("m3/h").Cells(topRow+
1, 14) =A dopterHYSYS.StdLiqVolFlow.GetValue("m3/h").Cells(topRow+2, 14) =B oilerBottomOutHYSYS.StdLiq
3, 14) =B oilerTopOutHYSYS.StdLiqVolFlow.GetValue("m3/h").Cells(topRow + 4, 14) =C
rystallizationInHYSYS.StdLiqVolFlow.GetValue("m3/h").Cells(topRow+5, 14) =C rystallizedRecycleHYSYS.Std
6, 14) =D ecanterOutHYSYS.StdLiqVolFlow.GetValue("m3/h").Cells(topRow + 7, 14) =D
istillationInHYSYS.StdLiqVolFlow.GetValue("m3/h").Cells(topRow+8, 14) =D istillationLiqOutHYSYS.StdLiqV
9, 14) =D istillationVapOutHYSYS.StdLiqVolFlow.GetValue("m3/h").Cells(topRow + 10, 14) =D
istVapOutXHYSYS.StdLiqVolFlow.GetValue("m3/h").Cells(topRow+11, 14) =E xternalOutHYSYS.StdLiqVolFlo
12, 14) =E xtractedBzAHYSYS.StdLiqVolFlow.GetValue("m3/h").Cells(topRow + 13, 14) =E
xtractedOthersHYSYS.StdLiqVolFlow.GetValue("m3/h").Cells(topRow+14, 14) =E xtractedWaterHYSYS.StdLiq
15, 14) =F eedHYSYS.StdLiqVolFlow.GetValue("m3/h").Cells(topRow+16, 14) =M ixerInHYSYS.StdLiqVolFlow.
17, 14) =P ostDistillationVapOutHYSYS.StdLiqVolFlow.GetValue("m3/h").Cells(topRow +
18, 14) =P ostDistVapOutXHYSYS.StdLiqVolFlow.GetValue("m3/h").Cells(topRow + 19, 14) =P
ostReactorOutHYSYS.StdLiqVolFlow.GetValue("m3/h").Cells(topRow+20, 14) =P reAdopterHYSYS.StdLiqVolFl
21, 14) =P reCrystallizedRecycleHYSYS.StdLiqVolFlow.GetValue("m3/h").Cells(topRow +
22, 14) =P reReactorInHYSYS.StdLiqVolFlow.GetValue("m3/h").Cells(topRow + 23, 14) =P
roductHYSYS.StdLiqVolFlow.GetValue("m3/h").Cells(topRow+24, 14) =P umpInHYSYS.StdLiqVolFlow.GetValu
25, 14) =R eactorInHYSYS.StdLiqVolFlow.GetValue("m3/h").Cells(topRow + 26, 14) =R
eactorOutHYSYS.StdLiqVolFlow.GetValue("m3/h").Cells(topRow+27, 14) =V alveOutHYSYS.StdLiqVolFlow.Get
.Cells(topRow + 0, 15) = AdditionalAirHYSYS.HeatFlow.GetValue("GJ/h").Cells(topRow +
1, 15) =A dopterHYSYS.HeatFlow.GetValue("GJ/h").Cells(topRow+2, 15) =B oilerBottomOutHYSYS.HeatFlow.C
3, 15) =B oilerTopOutHYSYS.HeatFlow.GetValue("GJ/h").Cells(topRow+4, 15) =C rystallizationInHYSYS.Heat
5, 15) =C rystallizedRecycleHYSYS.HeatFlow.GetValue("GJ/h").Cells(topRow + 6, 15) =D
ecanterOutHYSYS.HeatFlow.GetValue("GJ/h").Cells(topRow+7, 15) =D istillationInHYSYS.HeatFlow.GetValue
8, 15) =D istillationLiqOutHYSYS.HeatFlow.GetValue("GJ/h").Cells(topRow + 9, 15) =D
istillationVapOutHYSYS.HeatFlow.GetValue("GJ/h").Cells(topRow+10, 15) =D istVapOutXHYSYS.HeatFlow.C
11, 15) =E xternalOutHYSYS.HeatFlow.GetValue("GJ/h").Cells(topRow+12, 15) =E xtractedBzAHYSYS.HeatFl
13, 15) =E xtractedOthersHYSYS.HeatFlow.GetValue("GJ/h").Cells(topRow + 14, 15) =E
xtractedWaterHYSYS.HeatFlow.GetValue("GJ/h").Cells(topRow+15, 15) =F eedHYSYS.HeatFlow.GetValue("G
16, 15) =M ixerInHYSYS.HeatFlow.GetValue("GJ/h").Cells(topRow+17, 15) =P ostDistillationVapOutHYSYS.H
18, 15) =P ostDistVapOutXHYSYS.HeatFlow.GetValue("GJ/h").Cells(topRow + 19, 15) =P
ostReactorOutHYSYS.HeatFlow.GetValue("GJ/h").Cells(topRow+20, 15) =P reAdopterHYSYS.HeatFlow.GetVal
21, 15) =P reCrystallizedRecycleHYSYS.HeatFlow.GetValue("GJ/h").Cells(topRow + 22, 15) =P
reReactorInHYSYS.HeatFlow.GetValue("GJ/h").Cells(topRow+23, 15) =P roductHYSYS.HeatFlow.GetValue("G
24, 15) =P umpInHYSYS.HeatFlow.GetValue("GJ/h").Cells(topRow+25, 15) =R eactorInHYSYS.HeatFlow.GetV
26, 15) =R eactorOutHYSYS.HeatFlow.GetValue("GJ/h").Cells(topRow+27, 15) =V alveOutHYSYS.HeatFlow.Get
.Cells(10, 4) = reactorErr .Cells(11, 4) = crystallizerErr
.Cells(topRow + 47, 6) = CondenserHYSYS.HeatFlow.GetValue("MJ/h") .Cells(topRow + 47, 7) =

ToCond.Temperature.GetValue("C") .Cells(topRow + 47, 8) = Reflux.Temperature.GetValue("C")

.Cells(topRow + 48, 6) = $_{D}istVapOutExpandHYSYS.HeatFlow.GetValue("MJ/h").Cells(topRow+48,7) =_{D} istillationVapOutHYSYS.Temperature.GetValue("C").Cells(topRow + 48,8) =_{P} ostDistillationVapOutHYSYS.Temperature.GetValue("C")$

.Cells(topRow + 49, 6) = PowerOfPumpHYSYS.HeatFlow.GetValue("MJ/h") .Cells(topRow + 49, 7) = PumpInHYSYS.Temperature.GetValue("C") .Cells(topRow + 49, 8) = PreReactorInHYSYS.Temperature.GetValue("C")

.Cells(topRow + 50, 6) = $_{P}reCrystallizedRecycleHeaterHYSYS.HeatFlow.GetValue("MJ/h").Cells(topRow+50,7) =_{P} reCrystallizedRecycleHYSYS.Temperature.GetValue("C").Cells(topRow + 50,8) =_{C} rystallizedRecycleHYSYS.Temperature.GetValue("C")$

.Cells(topRow + 51, 6) = PreReactorInHeaterHYSYS.HeatFlow.GetValue("MJ/h") .Cells(topRow + 51, 7) = PreReactorInHYSYS.Temperature.GetValue("C") .Cells(topRow + 51, 8) = ReactorInHYSYS.Temperature.GetValue("C")

.Cells(topRow + 52, 6) = ValveOutCoolerHYSYS.HeatFlow.GetValue("MJ/h") .Cells(topRow + 52, 7) = DistillationLiqOutHYSYS.Temperature.GetValue("C") .Cells(topRow + 52, 8) = ValveOutHYSYS.Temperature.GetValue("C")

' .Cells(topRow + 53, 6) = ReactorOutCoolerHYSYS.HeatFlow.GetValue("MJ/h") ' .Cells(topRow + 53, 7) = ReactorOutHYSYS.Temperature.GetValue("C") ' .Cells(topRow + 53, 8) = DistillationInHYSYS.Temperature.GetValue("C")

.Cells(topRow + 54, 6) = ReboilerHYSYS.HeatFlow.GetValue("MJ/h") .Cells(topRow + 54, 8) = BoilUp.Temperature.GetValue("C") .Cells(topRow + 54, 7) = ToReb.Temperature.GetValue("C")

.Cells(topRow + 55, 6) = -((.Cells(topRow + 25, 15).Value + $.Cells(topRow + 28,15).Value + .Cells(topRow+30,15).Value)_{-}(.Cells(topRow+26).Value+.Cells(topRow+29,15).Value))_{*}1000.Cells(topRow+55,7) = reactorTemp - 273.15.Cells(topRow + 55,8) = reactorTemp - 273.15 + 0.1$

' .Cells(topRow + 60, 6) = ReactorOutPumpHYSYS.HeatFlow.GetValue("MJ/h") .Cells(topRow + 60, 6) = 0 .Cells(topRow + 60, 7) = PostReactorOutHYSYS.Temperature.GetValue("C") .Cells(topRow + 60, 8) = DistillationInHYSYS.Temperature.GetValue("C")

.Cells(topRow + 61, 6) = DistVapOutCoolerHYSYS.HeatFlow.GetValue("MJ/h") .Cells(topRow + 61, 7) = DistVapOutXHYSYS.Temperature.GetValue("C") .Cells(topRow + 61, 8) = PostDistVapOutXHYSYS.Temperature.GetValue("C")

.Cells(topRow + 70, 6) = ExternalOutCoolerHYSYS.HeatFlow.GetValue("MJ/h")

End With '————————————————————

'———————————————————— Dim CrystallizationInEnthalpy As Double '[MJ/h] Dim ProductEnthalpy As Double '[MJ/h] Dim PreCrystallizedRecycleEnthalpy As Double '[MJ/h]

CrystallizationInEnthalpy = CrystallizationInHYSYS. $_{E}nthalpyEstimate.GetValue("MJ/kgmole")_{*C} rystallizationInHYSYS.molarFlow.GetValue("kgmole/h")/1000ProductEnthalpy = ProductHYSYS.EnthalpyEstiroductHYSYS.molarFlow.GetValue("kgmole/h")/1000PreCrystallizedRecycleEnthalpy = PreCrystallizedRecycleHreCrystallizedRecycleHYSYS.molarFlow.GetValue("kgmole/h")_{/}1000$

With resultSheet .Cells(55, 27) = PowerOfPumpHYSYS.Power.GetValue("kW") .Cells(55, 25) = ExternalOutCoolerHYSYS.HeatFlow.GetValue("MJ/h") .Cells(55, 35) = DistVapOutExpandHYSYS.Power.GetValue("kW")

.Cells(topRow + 59, 6).Value = -(ProductEnthalpy + $_{P}reCrystallizedRecycleEnthalpy - {}_{C}rystallizationInEnthalpy) * 1000.Cells(topRow + 59, 8).Value = crystallizerTempEndWith$

Call RunPython("import heatexchange; heatexchange.main()") ' ThisWorkbook.Save '————————————————————————

Debug.Print "completed" End Sub

[caption=抽出塔] Option Explicit

Sub main(outletFlow As Variant, $_{B}zAFlowAsVariant,{}_{O}thersFlowAsVariant,{}_{W}aterFlowAsVariant,{}_{T}AsDouble$)

Dim toluene As Double '[kmol/h] Dim balc As Double '[kmol/h] Dim bald As Double '[kmol/h] Dim bza As Double '[kmol/h] Dim h2o As Double '[kmol/h] Dim Nitrogen As Double '[kmol/h] Dim Oxygen As Double '[kmol/h] Dim co As Double '[kmol/h] Dim co2 As Double '[kmol/h] Dim HotWater As Double '[kmol/h]

Dim Csat As Double ' 水単位質量当たりの安息香酸の溶解度 [BzA-g/ H2O-g]

' 蒸留塔出口の成分別モル数を求める. toluene = outletFlow(0) balc = outletFlow(1) bald = outletFlow(2) bza = outletFlow(3) h2o = outletFlow(4) Nitrogen = outletFlow(5) Oxygen = outletFlow(6) co = outletFlow(7) co2 = outletFlow(8)

' 安息香酸のモル量すべてが溶ける水の量を溶解度と温度から求める. Csat = $_{(}0.0000203 * T^4 + 0.000297 * T^3 + 0.047 * T^2 + 1.43 * T + 24.71)/{}_{1}000HotWater = ((bza * BzAMolarMass * 1000)/Csat)/WaterMolarMass/1000$

' 抽出器で，100BzAFlow(3) = bza

OthersFlow(0) = toluene OthersFlow(1) = balc OthersFlow(2) = bald OthersFlow(4) = h2o OthersFlow(5) = Nitrogen OthersFlow(6) = Oxygen OthersFlow(7) = co OthersFlow(8) = co2

WaterFlow(4) = HotWater

End Sub

[caption=晶析器] Option Explicit

'————————変数定義————————————————————————-

Private Const j$_{p}ow = 1.78PrivateConstb_{p}ow = 1.2PrivateConstg_{p}ow = 0.44PrivateConstR = 8.314PrivateConstkv = 0.1'$結晶体積形状係数

Private kb As Double ' 核発生速度 Private kg As Double ' 結晶成長速度 Private kg0 As Double ' 結晶成長頻度因子 Private Eg As Double ' 結晶成長活性化エネルギー Private rhoc As Double ' 結晶密度 Private GrowthRate As Double ' 結晶成長速度定数 Private T$_{M}SMPRAsDouble'$連続晶析器内温度 $PrivateT_{M}SMPR_{K}AsDouble'$連続晶析器内温度 $PrivateC_{F}eedAsDouble'Feed$ 液 Bac 濃度 $PrivateCsatAsDouble'Bac$ 飽和濃度 $Privatedelta_{C}AsDouble'$過飽和度 $PrivateMtAsDouble'$懸濁密度 (体積当たり結晶生産速度)$PrivateB0AsDouble'$一次核発生速度 $PrivaterhoLAsDouble'$液溶媒密度 $Privatetau_{M}SMPRAsDouble'$平均滞留時間 $Privaten0AsDouble'$個数密度 $PrivateVol_{M}SMPRAsDouble'$晶析装置液体積 $Privatev0_{M}SMPRAsDouble'Feed$ 液体積流量 $PrivateYcAsDouble'$全体結晶転化率 $PrivateYcmaxAsDouble'$可能結晶転化率

Function MSMPR(T$_{i}nputAsDouble,{}_{C}Feed_{i}nputAsDouble,{}_{v}0_{i}nputAsDouble,{}_{V}ol_{i}nputAsDouble,{}_{B}zAMolarFlowAsI$

' 諸条件から結晶生産量を求めます.

'————————操作条件入力———————————————————— T$_{M}SMPR = T_{i}nput'[℃]C_{F}eed = C_{F}eed_{i}nput'[g/g]v0_{M}SMPR = v0_{i}nput'[m3/h]Vol_{M}SMPR = Vol_{i}nput'[m3]$

'————————パラメータ入力————————————————————- Eg = 40.05 '[kJ/mol] kg0 = 10600000 '[μm/s] kb = 9160000000000 '[(/m3 s)(g/mL)$^{j}(BzA-g/H2O-g)^{b}]rhoc = 1.32'[g/cm3]rhoL = 0.998'[g/mL]$

’————諸量計算———————————————————- $\mathrm{tau}_M SMPR = Vol_M SMPR / v0_M SMPR *$ $60'[min] Csat = (0.0000203 * T_M SMPR^4 + 0.000297 * T_M SMPR^3 +_0 .047 * T_M SMPR^2 + 1.43 * T_M SMPR + 24.71)/1000'[g/g] T_M SMPR_K = T_M SMPR + 273.15'[K] kg = kg0 * Exp(-(Eg * 1000)/(R * T_M SMPR_K))'[\mu\ m/s]$

’————-晶析器内濃度 C の決定 (濃度誤差の二分法)———————————-

Dim C1 As Double, C2 As Double, C3 As Double ’ 仮定晶析器内濃度 Dim Cnew1 As Double, Cnew2 As Double, Cnew3 As Double ’ 更新濃度 Dim err1 As Double, err2 As Double, err3 As Double ’ 誤差 Dim counter As Integer ’ 計算回数 Dim eps As Double ’ 収束半径

eps = 0.0000001 counter = 0 C1 = Csat * 1.00000001 ’ 下限 C2 = $\mathrm{C}_F eed * 0.99999999'$上限

Call calc(C1, Cnew1, err1) ’ 小さい濃度 C1 での誤差検出 Call calc(C2, Cnew2, err2) ’ 大きい濃度 C2 での誤差検出

Do While (Abs((C1 - C2) / C1) ¿ eps)

C3 = (C1 + C2) / 2 Call calc(C3, Cnew3, err3) ’ 中間濃度 C3 での誤差検出’ 更新 If err1 * err3 ¿= 0 Then C1 = C3 err1 = err3 Else C2 = C3 err2 = err3 End If counter = counter + 1 Loop

’————出 力 部———————————————————- $\mathrm{Yc} = (\mathrm{C}_F eed - C3)/C_F eed * 100'[wt Ycmax = (C_F eed - C3)/(C_F eed - Csat) * 100'[wt$

MSMPR = BzAMolarFlow * Yc * Ycmax / 10000 ’MSMPR = $(\mathrm{C}_F eed - C3) * rhoL * v0_M SMPR * 1000'[kg/h] EndFunction$

Sub calc(ByRef C As Double, ByRef Cnew As Double, ByRef err As Double) ’ 計算ユニットです

Dim Mtnew As Double

Mt = $(\mathrm{C}_F eed - C) * rhoL'$定義式 $[g/mL] delta_C = C - Csat'$定義式 $[g/g] GrowthRate = kg * delta_{Cp}^{g} ow'$実験式 $[\mu\ m/s] B0 = kb * Mt_p^j ow * delta_{Cp}^{b} ow'$実験式 $[/m3s] n0 = B0/GrowthRate'$定義式 $[m3\ \mu\ m] Mtnew = 6 * kv * rhoc * (n0 * 1000000) *_((GrowthRate * 0.000001) * (tau_M SMPR * 60))^{4'}$個数収支式 $[g/mL] Cnew = C_F eed - Mtnew/rhoL'$個数収支式 $[g/g] err = C - Cnew'$誤差

End Sub

[caption=関数群] Option Explicit

Sub $\mathrm{solve}_k(TempAsDouble, kAsVariant)$

Dim i As Integer

Dim a As Variant Dim E As Variant Dim R As Double

a = Array(Exp(20.634), Exp(17.928), Exp(15.4), Exp(19.698)) E = Array(81.389, 69.53, 56.987, 71.442) k = Array(0, 0, 0, 0) R = 8.314

For i = 0 To 3 k(i) = a(i) * Exp(-E(i) * 1000 / R / Temp) Next i

End Sub

Sub solveX(ByRef F As Variant, ByRef x As Variant)

Dim i As Integer Dim sum As Long

sum = sumOfFlow(F)

For i = 0 To 8 x(i) = F(i) / sum Next i

End Sub

Function sumOfFlow(F As Variant) As Double

Dim i As Integer

sumOfFlow = 0

For i = 0 To 8 sumOfFlow = sumOfFlow + F(i) Next i

End Function

Sub solveComponentVolumeFlow(molarFlow As Variant, volumeFlow As Variant)

Dim i As Integer Dim param As Variant

' [m3/mol] param = Array(0.10550147646805, 0.10002241059429, 0.101176788713442, $_0$.112399721649694, 1.7752240578802, 0, 0, 0, 5.32284437764362$E-02$)

For i = 0 To 8 volumeFlow(i) = molarFlow(i) * param(i) Next

End Sub

[caption=最適化を行うコード] Option Explicit

Sub gridrun()

Dim topRow1 As Integer Dim topRow2 As Integer

Dim leftColumn1 As Integer Dim leftColumn2 As Integer

Dim resultSheet As Worksheet Dim writeSheet As Worksheet Dim saveSheet As Worksheet Set resultSheet = ThisWorkbook.Sheets(1) Set writeSheet = ThisWorkbook.Sheets(4) Set saveSheet = ThisWorkbook.Sheets(5)

Dim crVolume As Double Dim reVolume As Double

Dim i As Integer Dim j As Integer Dim p As Integer Dim l As Integer

topRow1 = 8 leftColumn1 = 4

topRow2 = 3 leftColumn2 = 2

For i = 1 To 1 j = 0

Call process$_a ll.main$

With writeSheet .Cells(topRow1 + j, leftColumn1 + i) = resultSheet.Cells(28, 22)

If (resultSheet.Cells(10, 4) ¿ 0.001) Then If (resultSheet.Cells(11, 4) ¿ 0.001) Then .Cells(topRow1 + j, leftColumn1 + i).Interior.Color = $_R GB(255, 0, 0) Else.Cells(topRow1 + j, leftColumn1 + i).Interior.Color =_R GB(0, 255, 0) EndIf ElseIf (resultSheet.Cells(11, 4) > 0.001) Then.Cells(topRow1 + j, leftColumn1 + i).Interior.Color =_R GB(0, 0, 255) EndIf EndIf EndWith Next i$

End Sub