Département : Chimie Appliquée et Environnement

Filière Ingénieur : Procédés et Ingénierie Chimique

Année Universitaire : 2025-2026

Devoir de modélisation et simulation

Rédigé par :
NIAPA Annicque Orlande
Wend-Kouni

Encadré par :
Pr. BAKHER

Table des matières

Introduction

La distillation de mélanges multicomposants est une opération utilisée pour séparer des mélanges complexes. Elle nécessite des méthodes de calcul plus sophistiquées pour tenir compte des interactions entre tous les composés.

L'objectif de ce projet est de comprendre les méthodes de calcul utilisées dans la distillation multicomposants, d'implémenter les algorithmes en Python avec les bibliothèques `thermo`, `scipy`, et `plotly`, de créer des visualisations interactives des résultats et de dimensionner une colonne industrielle complète.

Nous allons dans les lignes suivantes, analyser le code mis à notre disposition et donner le rôle des classes et fonctions utilisées.

1 Concepts théoriques utilisées dans la distillation multicomposants

Equilibre liquide-vapeur

Loi de Raoult modifiée :

$$y_i P = \gamma_i x_i P_{sat}(T) \tag{1}$$

Le coefficient de partage K_i :

$$K_i = \frac{y_i}{x_i} = \frac{\gamma_i P_{sat}(T)}{P} \tag{2}$$

Contrainte de normalisation :

$$\sum_{i=1}^{n} x_i = 1 \quad ; \quad \sum_{i=1}^{n} y_i = 1 \tag{3}$$

Volatilité relative : Elle permet de connaitre l'ordre de volatilité des composants du mélange. Le composant le plus léger a l'ordre de volatilité la plus grande avec r le composé de référence.

$$\alpha_{i,r} = \frac{K_i}{K_r} = \frac{P_{sat,i}(T)}{P_{sat,r}(T)} \tag{4}$$

La température de bulle : Température à laquelle la première bulle de vapeur se forme.

Température de rosée : Température à laquelle la première goutte de liquide se forme.

Méthodes de calculs

— **Méthode de Fenske :** permet de déterminer le nombre minimum de plateaux N_{min}.

— **Méthode d'Underwood :** permet de calculer le reflux minimum R_{min} pour une séparation donnée.
— **Corrélation de Gilliland :** c'est une corrélation empirique reliant le nombre de plateaux au reflux.
— **Équation de Kirkbride :** permet de déterminer la position optimale du plateau d'alimentation.
— **Méthode Rigoureuse (Équations MESH) :** Le système MESH est la méthode rigoureuse standard pour la simulation de colonnes de distillation. Il résout simultanément les équations de :
  — **M**aterial balance (bilans matières)
  — **E**quilibrium (équilibre thermodynamique)
  — **S**ummation (contraintes de normalisation)
  — **H**eat balance (bilans énergétiques)

2 Analyse du code distillation_multicomposents.py

2.1 Rôle des bibliothèques Python utilisées

Dans ce projet de simulation thermodynamique, plusieurs bibliothèques Python spécialisées ont été utilisées pour assurer le calcul, la résolution des équations et la visualisation des résultats.

— **Numpy :** constitue la base de tous les calculs numériques réalisés dans ce projet. Elle permet la manipulation rapide de tableaux et de matrices, indispensables pour les bilans de matière, les calculs de compositions, les opérations matricielles et l'évaluation itérative des modèles thermodynamiques.
— **Matplotlib :** est utilisée pour créer des graphiques 2D classiques et permet de visualiser les courbes obtenues lors de la simulation : évolution des compositions, profils thermiques, convergence, représentations des équilibres.
— **SciPy :** fournit des outils mathématiques avancés permettant de résoudre les équations qui interviennent dans la modélisation d'une distillation.
  — `fsolve` : utilisée pour trouver les températures d'équilibre (bubble/dew).
  — `brentq` : pour trouver les racines (méthode d'Underwood).
  — `minimize` : disponible pour les optimisations éventuelles.
  — `solve_banded` : permet de résoudre efficacement des systèmes d'équations de type bande, rencontrés dans les équations MESH.
— **thermo** (`thermo.chemical.Chemical`, `ChemicalConstantsPackage`, `PRMIX`, `CEOSLiquid`, `CEOSGas`) : La bibliothèque thermo constitue le cœur de la partie thermodynamique du projet. Elle permet d'accéder aux propriétés physiques des composés (T_c, P_c, masse molaire, facteur acentrique) ; d'utiliser des équations d'état (comme Peng–Robinson, Raoult) pour modéliser les phases liquide et vapeur ; de calculer les coefficients de fuga-

cité, les constantes d'équilibre (K-values), ainsi que les équilibres liquide-vapeur.

— **warnings :** est utilisé pour filtrer et masquer certains messages d'avertissement générés lors des calculs, améliorant la lisibilité des résultats.

— **plotly** : utilisée pour produire des visualisations interactives (courbes zoomables, graphiques multi-composants, 3D).

2.2 Analyse des classes et méthodes

Class compound

Elle représente un composé chimique unique avec ses propriétés thermodynamiques. Son role est d'encapsuler les données et methodes d'un composé chimique et de récupérer automatiquement les propriétés depuis la base de données. Les methodes principales utilisées sont :

— `__init__(self, name)` : charge les propriétés depuis la base de données.

— `vapor_pressure(T)` : calcule la pression de vapeur saturante à une température donnée.

— `K_value(T, P)` : calcule le coefficient de partage $K = P_{sat}/P$.

— `enthalpy_liquid(T)` : enthalpie du liquide.

— `enthalpy_vapor(T)` : enthalpie de la vapeur.

Class ThermodynamicPackage

Cette classe agrège les propriétés thermodynamiques de tous les composés d'un mélange et calcule les équilibres liquide-vapeur. Son objectif est de gérer une liste de composés, effectuer les calculs thermodynamiques au niveau du mélange et de résoudre les équilibres de phases. Les methodes principales utilisées sont :

— `K_values()` : Calcule les coefficients d'équilibre K de tous les composés.

— `relative_volatilities(T, P)` : Compare les K entre composés.

— `bubble_temperature(p, x)` : Trouve T où le liquide commence à bouillir.

— `dew_temperature(p, y)` : Trouve T où la vapeur commence à condenser.

— `mixture_enthalpy_liquid` / `_vapor` : Calcule les enthalpies moyennes.

Class ShortcutDistillation

Cette classe dimensionne une colonne de distillation en utilisant des méthodes simplifiées rapides. Son rôle principal est de prédire les débits de distillat et de résidu, les compositions des produits, le nombre de plateaux réels et théoriques, la position du plateau d'alimentation ainsi que les débits internes. Les methodes utilisées sont :

— `_identify_key_components()` : Identifie les composés clés (LK, HK).

— `material_balance` : Calcule débits D, B et compositions x_D, x_B.

— `fenske_equation()` : Calcule N_{min}.

— `underwood_method()` : Calcule R_{min}.
— `gilliland_correlation()` : Estime N théorique pour un reflux R donné.
— `kirkbride_equation()` : Détermine la position du plateau d'alimentation.

3 Analyse du code visualisation.py

3.1 Analyse des classes et methodes

Class DistillationVisualizer : Cette classe crée des visualisations statiques(matplotlib) et interactives(plotly) des résultats de dimensionnement.

— `plot_material_balance()` : graphiques débits et compositions.
— `plot_shortcut_results()` : dashboard complet (Fenske, Underwood, Gilliland...).
— `_draw_column_schematic()` : schéma interne de la colonne.
— `plot_composition_profiles_matplotlib/plotly()` : profils liquide/vapeur.
— `plot_temperature_profile()` : Profil de température.

3.2 Les fonctions

`print_design_summary()` : son rôle est d'afficher le résumé formaté en console(bilans, compositions, paramètres, débit internes).

4 Analyse du code exemple_btx.py

Ce code fournit un exemple de dimensionnement pour le mélange BTX (benzène, toluène, o-xylène) en utilisant les méthodes « shortcut » (Fenske, Underwood, Gilliland, Kirkbride), estimer profils et énergie, puis générer des visualisations.

4.1 Les fonctions

— `exemple_btx_complet` : est utilisée pour une exécution complète (définition, dimensionnement, estimations, visualisations, bilan et énergie)
— `etude_parametrique_reflux()` : son objectif est d'étudier l'effet du rapport de reflux (R) sur le nombre de plateaux requis pour la séparation BTX et proposer un "point optimal" simple en combinant compromis entre coût (R) et taille (N).

4.2 Etapes du code

1. **Définition du système** : Charger les composés, agréger les données, fixer les conditions (P, F, z_F).

2. **Dimensionnement par méthodes simplifiées** : Instanciation, spécification de la séparation, exécution des calculs (Fenske, Underwood...), récupération des résultats.
3. **Estimation des profils** : Son objectif est d'estimer rapidement les profils de composition (x, y) et de température le long des plateaux pour visualisation et calculs approximatifs. Interpolation linéaire et calcul de T_{bubble}.
4. **Visualisation** : Son rôle est de produire des graphiques clairs des résultats (bilans, schéma colonne, profils composition/température, corrélations) pour une inspection rapide. Les outils utilisés sont matplotlib et pyplot. Les composants clés sont : schéma colonne, courbe Gilliland (N vs R), barres débits/compositions, profils x/y par plateau, profil de température. .
5. **Analyse des résultats** : Son rôle est de présenter la distribution des composés et vérifier la cohérence des bilans matière après le dimensionnement. Pour chaque composé i on calcule apport en alimentation, le débit dans le distillat, le débit dans le résidu et la récupération dans D. Il y a également une vérification de l'erreur globale. .
6. **Estimation énergétique** : Son objectif est d'estimer les besoins en chauffage (rebouilleur) et refroidissement (condenseur) et déduire consommation de vapeur. Il calcul également les enthalpies : pour tête et fond de colonne..

5 Résultats obtenus avec Aspen plus

Spécifications : Alimentation : 100 kmol/h, Composition : 33.3% benzène, 33.3% toluène, 33.4% xylène, Pression : 1.013 bar. Objectif : 95% benzène dans distillat, 95% toluène+xylène dans résidu.

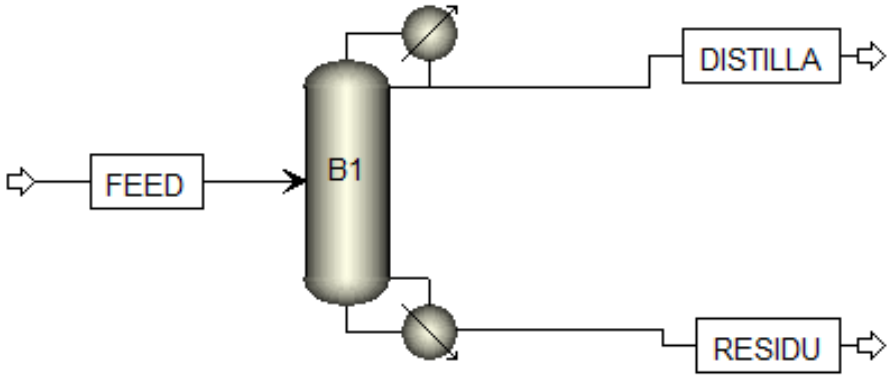

FIGURE 1 – Résultats Aspen Plus

DSTWU

Light key component recovery	0,95
Heavy key component recovery	0,05
Distillate vapor fraction	0
Minimum reflux ratio	1,31174
Actual reflux ratio	1,70526
Minimum number of stages	6,74278
Number of actual stage	13,2201
Feed stage	7,41553
Number of actual stage above feed	6,41553
Distillate temperature [C]	81,124
Bottom temperature [C]	122,35
Distillate to feed fraction	0,333015

FIGURE 2 – Schéma du procédé Aspen Plus

6 Tableau comparatif

Tableau récapitulatif des résultats du dimensionnement réalisé avec Python comparé à Aspen Plus.

Paramètre	**Python**	**Aspen Plus**	**Ecart (%)**
N_{min} (Fenske)	3,22	6,74	52,00
R_{min} (Underwood)	0,50	1,31	61,83
R opératoire ($1.3 \times R_{min}$)	0,65	1,71	61,98
N théorique (Gilliland)	4,51	-	-
Efficacité	70%	-	-
N réel	7	13	46,15
Plateau d'alimentation (Kirkbride)	5	7,43	32,70
Débit distillat (kmol/h)	43,91	33,30	31,86
Débit résidu (kmol/h)	56,09	66,70	15,89

TABLE 1 – Comparaison des résultats Python vs Aspen Plus

Note : L'écart est calculé selon la formule :

$$\text{Ecart} = \frac{|\text{Valeur}_{Aspen} - \text{Valeur}_{Python}|}{\text{Valeur}_{Aspen}}$$

La comparaison montre un écart assez considérable entre les deux méthodes. Cela s'explique principalement par les différences au niveau des modèles thermodynamiques et d'hypothèses numériques.

Conclusion

L'utilisation combinée de Python et d'Aspen Plus dans ce projet a permis d'exploiter pleinement les avantages du calcul scientifique et de la simulation procédés. Aspen Plus a offert un environnement robuste pour la modélisation thermodynamique, la simulation des opérations unitaires et l'obtention de résultats fiables sur le comportement du système étudié. En parallèle, Python a renforcé la capacité d'analyse grâce à son excellente flexibilité, la possibilité d'automatiser des calculs, de traiter les données issues d'Aspen, et de visualiser les résultats de manière plus claire et personnalisée. Cela montre l'intérêt croissant de l'intégration du numérique dans le génie des procédés.