Reator Tubular Autotérmico Para Produção de Amônia

João Eduardo Levandoski, RA: 265243

17 de janeiro de 2021

A reação de síntese de amônia é limitada pelo equilíbrio e é demonstrada pela Equação 1.

$$N_2 + 3H_2 \leftrightarrow 2NH_3 \tag{1}$$

A conversão de amônia será obtida em um reator do tipo autotérmico, deste modo o reagente será utilizado como fluido refrigerante da própria reação.

Para obtenção dos perfis de é necessário o equacionamento dos balanços de massa e energia do sistema.

O balanço de massa será feito em função do produto que é a amônia NH_3, Equação 2

$$\frac{dF_A}{dV} = 2r \tag{2}$$

Como V é o volume da reação e temos a área do tubo que é A_c, pode-se escrever o BM em função do comprimento do reator, Equação 3.

$$\frac{dF_A}{dz} = 2A_c r \tag{3}$$

Assumindo a condição de gás ideal e com os dados fornecidos obtém-se a vazão molar com a Equação 4.

$$F_T = \frac{P_T \cdot Q_0}{RT_0} \tag{4}$$

Logo a vazão de amônia com base nas frações de entrada, dadas no enunciado, é obtida pela Equação 5.

$$F_{A0} = F_T x^0_{NH_3} \tag{5}$$

O enunciado forneceu os parâmetros γ e β que são parte integrantes do balanço de energia, sendo assim o balanço de energia será organizado de modo a poder utilizar eles. De modo geral o BE pode ser escrito como a Equação 6

$$Q\rho\hat{C}_p\frac{dT}{dV} = \dot{q}_{fluido} - \Delta H_R \cdot r \tag{6}$$

Sendo q o calor do fluido refrigerante, que pode ser escrito como a Equação 7, onde T_a é a temperatura do fluido refrigerante e T a do reator.

$$\dot{q}_{fluido} = \frac{2}{R}U(T_a - T) \tag{7}$$

Desde modo a juntando a Equação 6 com 7 e dividindo tudo por $Q\rho\hat{C}_p$, obtem-se a Equação

$$\frac{dT}{dV} = -\frac{\Delta H_R \cdot r}{Q\rho\hat{C}_p} + \frac{2U(T_a - T)}{RQ\rho\hat{C}_p} \tag{8}$$

Colocando em função de z, sabendo que $dV = A_c dz$ e $A_c = \pi R^2$, o balanço de energia para o reator fica como na Equação 9

$$\frac{dT}{dz} = -\frac{\Delta H_R A_c}{Q\rho\hat{C}_p}r + \frac{2U}{RQ\rho\hat{C}_p}(T_a - T) \tag{9}$$

Substituindo $\frac{2U}{RQ\rho\hat{C}_p}$ por γ e $\frac{\Delta H_R A_c}{Q\rho\hat{C}_p}$ por β temos a Equação 10, como BE final para o reator.

$$\frac{dT}{dz} = -\beta r + \gamma(T_a - T) \tag{10}$$

O balanço de energia para o fluido refrigerante pode ser escrito como a Equação 11

$$Q_a\rho_a\hat{C_{pa}}\frac{dT_a}{dV} = \dot{q}_{fluido} \tag{11}$$

Como o fluido refrigerante possui a mesma vazão e propriedades dos reagentes o termo $Q_a\rho_a\hat{C_{pa}}$ é igual a $Q\rho\hat{C}_p$, e se escrever em função do comprimento do reator e sabendo que $dV = A_c dz$ e $A_c = \pi R^2$, tem-se a Equação 12

$$\frac{dT_a}{dV} = \frac{2\pi RU}{Q\rho\hat{C}_p}(T_a - T) \tag{12}$$

Substituindo $\frac{2U}{RQ\rho\hat{C}_p}$ por γ o BE final para o fluido refrigerante é a Equação 13

$$\frac{dT_a}{dV} = \gamma(T_a - T) \tag{13}$$

Com os balanços de massa e energia, e o modelo cinético fornecido no enunciado Equação 14

$$-r = \frac{k_{-1}}{RT}\left(K^2\frac{P_{N_2}P_{H_2}^{3/2}}{P_{NH_3}} - \frac{P_{NH_3}}{P_{H_2}^{3/2}}\right) \tag{14}$$

A velocidade específica pode ser calculada com a Equação 15

$$k_{-1} = k_{-1_0}exp\left[\frac{-E_a}{R}\frac{1}{T}\right] \tag{15}$$

Constante de equilíbrio na temperatura de referência:

$$K(T_R) = exp\left[\frac{-\Delta G^o}{R}\frac{1}{T}\right] \tag{16}$$

Variação da constante de equilíbrio em função da temperatura:

$$K = K(T_R)exp\left[\frac{\Delta H_R}{R}\Big(\frac{1}{T_r} - \frac{1}{T}\Big)\right] \tag{17}$$

Vazões molares:

$$F_T = \sum F_i \tag{18}$$

Pressão parcial seguindo a lei de Raoult:

$$P_i = y_i P_T \tag{19}$$

Balanço estequiométrico:

$$F_{N_2} = F_{N_2}^0 - 1/2(F_{NH_3} - F_{NH_3}^0) \tag{20}$$

$$F_{H_2} = F_{H_2}^0 - 3/2(F_{NH_3} - F_{NH_3}^0) \tag{21}$$

Alguns dados não foram compatíveis (Q e ΔG^0) com a resolução correta, sendo assim para ser possível a resolução os dados utilizados foram:

- $P = 30,3975 \times 10^6 Pa$
- $L = 12 \;\; m$
- $A_c = 1 \;\; m^2$
- $Q = 0,16 \;\; m^3/s$
- $x_{NH_3}^0 = 0,015$
- $x_{N2}^0 = 0,985(1/4)$
- $x_{H2}^0 = 0,985(3/4)$

- $\gamma = 0,5 \quad m^{-1}$
- $\beta = -2,342 \quad m^2.K.s/mol$
- $k_{1_0} = 7,897 \times 10^{16} \quad Pa/s$
- $\frac{E}{R} = 2 \times 10^4 \quad K$
- $\Delta H_r = (-5,02 \times 10^4) \quad J/mol$
- $\Delta G^0 = (17,782 \times 10^3) \quad J/mol$

Com o uso da linguagem de programação Python, resolveu-se o sistema de EDOs acima demonstrado. O algoritmo desenvolvido pedia um chute de temperatura de entrada, o chute foi de 400 K, após isso foram feitas aproximações automáticas, após 192 iterações encontrou-se a T de entrada tal que a T_a fosse igual a 323 K.

A Temperatura inicial obtida foi:

$$T_a(0) = T(0) = 920,10K$$

Com a temperatura acima chegou a conversão:

$$X = 30,38\%$$

Os perfis obtidos para as temperatura foram os disponibilizados na Figura 1, os demais perfis de vazões e frações estão nas Figuras 2 e 3, respectivamente.

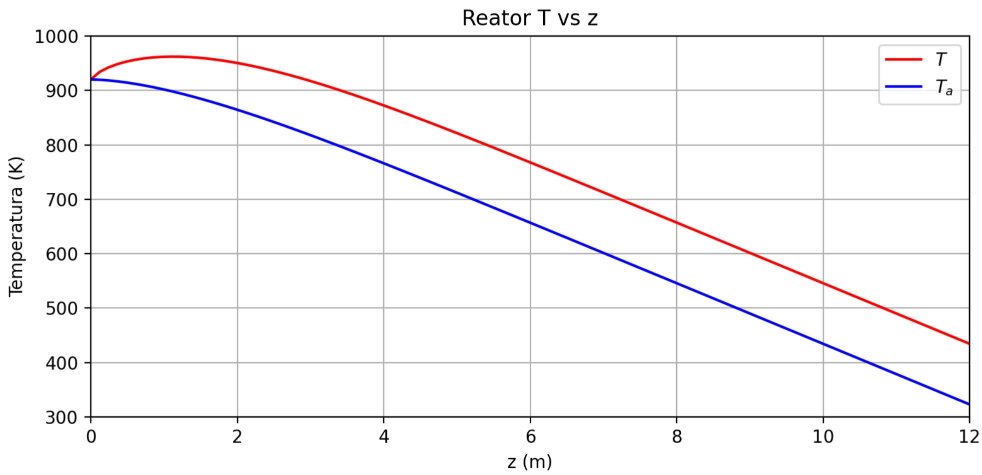

Figura 1 – Temperaturas versus z

Segue em anexo o algoritmo implementado em Python completo utilizado nos cálculos. Esse algoritmo pode ser executado no Google Colab online.

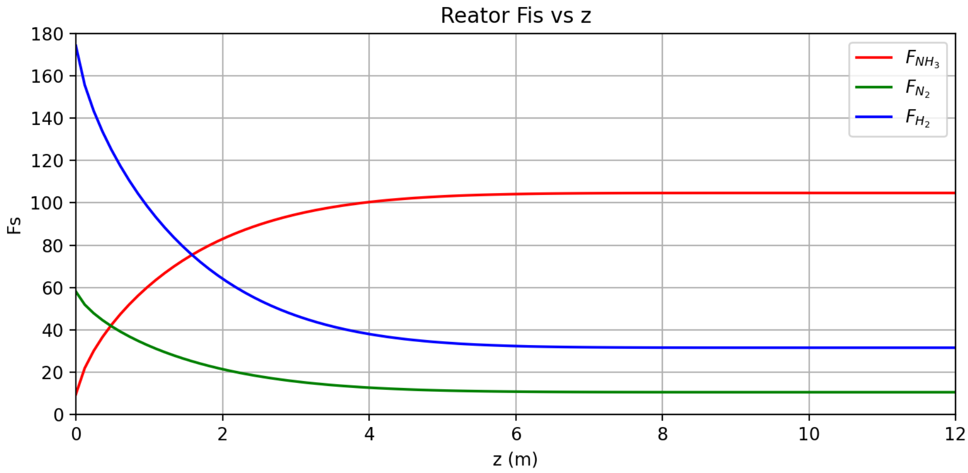

Figura 2 – Vazões versus z

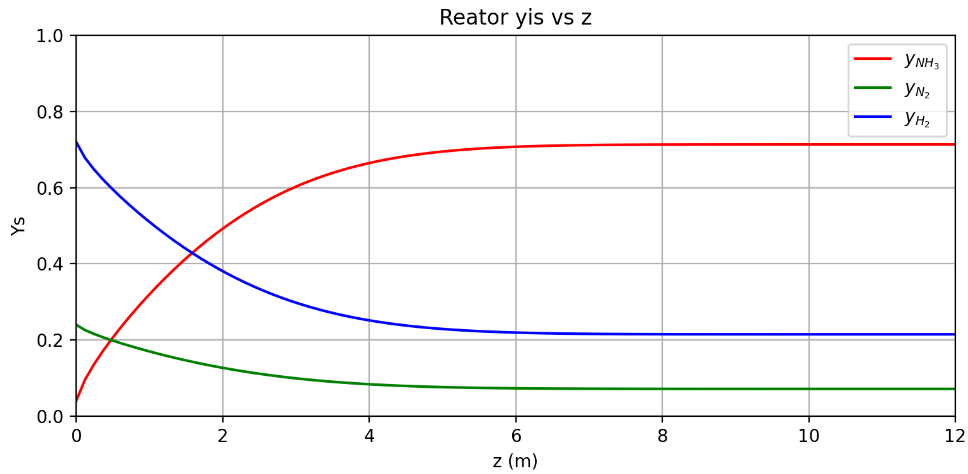

Figura 3 – Frações versus z

Anexos

```python
# -*- coding: utf-8 -*-
import numpy as np
from numpy import exp
import matplotlib.pyplot as plt
from scipy.integrate import odeint

def plots():
    plt.figure(dpi=100, figsize=(9, 4))
    plt.plot(z, T, 'r', label='$T$')
    plt.plot(z, Ta, 'b', label='$T_a$')
    plt.xlabel('z (m)')
    plt.ylabel('Temperatura (K)')
    plt.title('Reator T vs z')
    plt.xlim(0, L)
    plt.ylim(300, 1000)
    plt.legend()
```

```python
    plt.grid()
    plt.savefig('temp.png', dpi=200)
    plt.show()

    Fn =  Fnf - (1/2)*(Fa - Faf)
    Fh =  Fhf - (3/2)*(Fa - Faf)

    plt.figure(dpi=100, figsize=(9, 4))
    plt.plot(z, Fa, 'r', label='$F_{NH_3}$')
    plt.plot(z, Fn, 'g', label='$F_{N_2}$')
    plt.plot(z, Fh, 'b', label='$F_{H_2}$')
    plt.xlabel('z (m)')
    plt.ylabel('Fs')
    plt.title('Reator Fis vs z')
    plt.xlim(0, L)
    plt.ylim(0, 180)
    plt.legend()
    plt.grid()
    plt.savefig('Fis.png', dpi=200)
    plt.show()

    Ft = Fa + Fn + Fh
    ya = Fa/Ft
    yn = Fn/Ft
    yh = Fh/Ft

    plt.figure(dpi=100, figsize=(9, 4))
    plt.plot(z, ya, 'r', label='$y_{NH_3}$')
    plt.plot(z, yn, 'g', label='$y_{N_2}$')
    plt.plot(z, yh, 'b', label='$y_{H_2}$')
    plt.xlabel('z (m)')
    plt.ylabel('Ys')
    plt.title('Reator yis vs z')
    plt.xlim(0, L)
    plt.ylim(0, 1)
    plt.legend()
    plt.grid()
    plt.savefig('Yis.png', dpi=200)
    plt.show()

def edo(y, z):
    global X
    FA, T, Ta = y
    K298 = exp(-deltaG/Rg/T)
    K = K298*exp((deltaHr/Rg)*(1/Tr - 1/T))

    #Velocidade especifica
    k1 = k10*exp(-ER/T) #Pa/s

    #Determinacao da vazao molar inicial
    FT0 = v0*(P)/Rg/T0 #mol/s

    #Vaz o molar inicial de cada composto
    FAf = FT0*x0nh3 #mol/s
    FH2f = FT0*x0h2 #mol/s
    FN2f = FT0*x0n2 #mol/s

    #Vaz o molar de nitrog nio e hidrog nio com o decorrer da rea   o
    FN2 = FN2f - 0.5*(FA - FAf) #mol/s
    FH2 = FH2f - 1.5*(FA - FAf) #mol/s
```

```

    #Vaz o molar total com o decorrer da rea  o
    FT = FA + FN2 + FH2 #mol/s


    #Press o
    yA = FA/FT
    yN2 = FN2/FT
    yH2 = FH2/FT
    Pa = P*yA #Pa
    Ph2 = P*yH2 #Pa
    Pn2 = P*yN2 #Pa

    #Convers o de nitrog nio
    X = 1 - FN2/FN2f

    #Equa  o da taxa
    r = (k1/(Rg*T))*((K**2)*(((Pn2)*Ph2**1.5)/Pa)-Pa/(Ph2**1.5))

    #Balan o de massa
    dFAdz= 2*Ac*r # 2xAc

    #Balan o de energia
    dTdz = -beta*r + gama*(Ta - T) #K/m

    #Balan o de energia do fluido refrigerante
    dTadz= gama*(Ta - T) #K/m

    dydz = [dFAdz, dTdz, dTadz]
    return dydz

global X
#Dados fornecidos pelo enunciado
P = 300*101325 #Pa
L = 12 #m
Ac = 1 # m
v0 = 0.16
x0nh3 = 0.015
x0n2 = 0.985*(1/4)
x0h2 = 0.985*(3/4)
gama = 0.5 #1/m
beta = -2.342 #m Ks/mol
k10 = (7.794*10**11)*101325 #Pa/s
ER = (2*10**4) #K
deltaHr = (-1.2*10**4)*4.184 #J/mol
deltaG = (4.25*10**3)*4.184 #J/mol

#Constante universal dos gases
Rg = 8.3145 #J/mol/K

#Constante de equil brio
Tr = 298 #K

i = 0
Taf = 323 #323

chute = 400
while True and i<200:
    i+=1
    if i > 1:
```

```python
        Tfinal = Ta[-1]
        dif = (Tfinal - Taf)
        erro = dif/Taf
        print(f'i: {i} x: {X} Tentrada: {Tentrada} Tfinal:{Tfinal} delta: {dif} erro: {erro}')
        if abs(erro) < 1e-4:
            break
        if Tfinal > Taf:
            Tentrada = Tentrada + dif/30
        else:
            Tentrada = Tentrada - dif/30
    else:
        Tentrada = chute
    z = np.linspace(0, L, 100)

    T0 = Tentrada #K
    FT0 = v0*(P)/Rg/T0 #mol/s
    Fa0 = FT0*x0nh3
    # Na, T, Ta
    y0 = Fa0, Tentrada, Tentrada
    sol = odeint(edo, y0, z)
    Fa, T, Ta = sol[:, 0], sol[:, 1], sol[:, 2]

plots()
```

Listing 1 – Código para o cálculo