

**Simulação da injeção de bancos  
de água com polímeros na  
recuperação de petróleo.**

**José Roberto Nogueira**

# Simulação da injeção de bancos de água com polímeros na recuperação de petróleo.

Este exemplar corresponde a redação final da tese de doutorado devidamente corrigida e defendida por José Roberto Nogueira e aprovada pela comissão julgadora.

Campinas, 28 de Janeiro de 2000



---

Prof. Dra. Maria Cristina de Castro Cunha  
Orientadora

Banca Examinadora:

- 1- Prof. Dr. Antonio Claudio de França Corrêa
- 2- Prof. Dr. Dan Marchesin
- 3- Prof. Dr. José Luiz Boldrini
- 4- Prof. Dr. Milton da Costa Lopes Filho

Tese apresentada ao Instituto de Matemática, Estatística e Computação Científica, UNICAMP, como requisito necessário para obtenção do Título de Doutor em Matemática Aplicada.

UNICAMP  
N.º CHAMADA:  
V.º  
TOMADA DE  
PREMIO  
DATA  
N.º

FICHA CATALOGRÁFICA ELABORADA PELA  
BIBLIOTECA CENTRAL DA UNICAMP

N689s

Nogueira, José Roberto

Simulação da injeção de bancos de água com  
polímeros na recuperação do petróleo / José Roberto  
Nogueira. — Campinas, SP : [s.n.], 2000.

Orientador : Maria Cristina Cunha.

Tese (doutorado) - Universidade Estadual de  
Campinas, Instituto de Matemática, Estatística e Ciência  
da Computação.

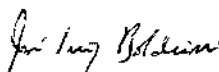
1. Riemann-Hilbert, Problemas de. 2. Recuperação  
secundária do petróleo. 3. Métodos de simulação.  
I. Cunha, Maria Cristina. II. Universidade Estadual de  
Campinas. Instituto de Matemática, Estatística e Ciência da  
Computação. III. Título.

**Tese de Doutorado defendida e aprovada em 10 de dezembro de 1999**


**Pela Banca Examinadora composta pelos Profs. Drs.**



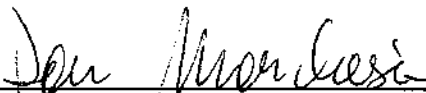
**Prof (a). Dr (a). MARIA CRISTINA DE CASTRO CUNHA**



**Prof (a)./ Dr (a). JOSÉ LUIZ BOLDRINI**



**Prof (a). Dr (a). MILTON DA COSTA LOPES FILHO**



**Prof (a). Dr (a). DAN MARCHESIN**



**Prof (a). Dr (a). ANTONIO CLAUDIO DE FRANÇA CORRÊA**

# Dedicatória

Dedico este trabalho à meus pais,  
minha esposa  
e meu filho.

## Agradecimentos

Agradeço inicialmente a DEUS, pelo dia, pela noite, pelas flores e, enfim pela vida.

Agradeço a todos os amigos do departamento de matemática da FCT-UNESP que sempre torceram pelo meu sucesso. Em especial aos Profs. Aylton, Suetonio e Gabriel.

Agradeço a todos os amigos que fiz no IMECC, alguns dos quais convivi muito. Em especial à Fátima, secretária da matemática aplicada, pelo carinho com que sempre me tratou.

Agradeço aos amigos Sidinei, Maria, Cris e Thiago que me deram acolhida e participaram de muitos momentos difíceis desta caminhada.

Agradeço aos amigos das quintas-feiras pela força, pelo incentivo e pela amizade, especialmente a Elza, Vera e Orlando.

Agradeço ao amigo Prof. Messias, pelo incentivo e por ter acreditado no meu trabalho.

Agradeço especialmente a minha orientadora, Cristina, pela orientação, pela amizade e pela paciência que teve durante estes anos para comigo.

Agradeço a meus pais, José e Cida pela formação e educação que me deram, por eles serem pessoas lindas e serem meus pais. Agradeço também a meus irmãos que sempre torceram por mim.

Agradeço também a Zélia e Braulio que mesmo estando longe estavam torcendo e rezando por mim. Agradeço também ao amigo “ véio “ vulgo Jorge Luis por ter participado e nos ajudado em alguns momentos difíceis.

Agradeço a minha mulher Socorro e ao meu filho Bruno por serem maravilhosos, por terem aturado minhas neuras durante este tempo todo e pelo grande amor que sinto por eles.

## Resumo

Neste trabalho abordamos o problema da injeção de bancos de água com polímeros na recuperação de petróleo. O modelo matemático consiste num sistema de leis de conservação com condições iniciais e de fronteira apropriadas. Utilizamos as soluções do problema de Riemann ( [ISA] e [JOH] ) associado a este sistema para propor um algoritmo que calcula os perfis de saturação de água  $s(x, t)$ . As condições de contorno são ditadas pelo tamanho dos bancos de água (ou água com polímero) e a concentração de polímero usada no poço de injeção. As dificuldades encontradas se devem ao tratamento das descontinuidades que surgem devido as interações entre rarefações e choques que se formam devido a solução do problema de Riemann e as descontinuidades nas condições de contorno. Um programa computacional foi elaborado para calcular as interações entre os choques, a solução do problema de Riemann e a solução do problema da injeção de bancos. Foram considerados os casos com e sem adsorção do polímero pelo meio poroso. Simulamos a recuperação de óleo em alguns casos onde variamos o tamanho dos bancos, a concentração de polímero além das condições iniciais nos reservatórios. Fizemos comparações entre o algoritmo proposto e um esquema numérico do tipo upwind.

## Abstract

In this presentation we work on the problem of the injection of water banks with polymers in enhanced oil recovery. The mathematical model consists of a system of conservation laws with appropriate initial conditions and of boundaries. We have used the solutions of the Riemann problem ( [ISA] and [JOH] ) associated to this system to propose an algorithm which calculates the profiles of saturation of water  $s(x,t)$ . The outlining conditions are given through the size of the water banks ( or water whit polymers ) and the polymer concentration used in the injector well. The difficulties found are due to the treatment of the discontinuities that appear due to the interaction among rarefaction and shocks that are formed because of the solution of the Riemann problem and the discontinuities in the outlining conditions. We have made program computer which calculates the interaction among the shocks, the solution of the Riemann problem and the solution of the banks injection problem. Cases with or without polymer adsorption by the porous environment have been considered. Some cases of recuperation of oil have been simulated whereas the size of the banks and polymer was varied, as well as the initial conditions of the reservoir. The proposed algorithm and numerical scheme have been compared.



# CONTEÚDO

<b>1</b>	<b>Introdução</b>	<b>4</b>
1.1	Revisão Bibliográfica: . . . . .	5
1.1.1	O Trabalho de Isaacson . . . . .	5
1.1.2	O Trabalho de Johansen & Winther . . . . .	5
1.1.3	Os Trabalhos de Barenblat & outros e Bedrikovetsky . . . . .	6
1.2	<b>Objetivos:</b> . . . . .	6
1.3	Organização do Trabalho: . . . . .	6
<b>2</b>	<b>O Problema de Riemann</b>	<b>8</b>
2.1	A injeção de água . . . . .	9
2.2	A injeção de polímero ( polymer-flooding ). . . . .	10
2.3	A solução do Problema de Riemann: sem adsorção. . . . .	11
2.3.1	CASO A : $c_d = c_e$ . . . . .	14
2.3.2	CASO B: $u_e \in L$ e $u_d \in L_1 \cup L_2 \cup L_3$ . . . . .	16
2.3.3	CASO C : $u_e \in R$ e $u_d \in R_1 \cup R_2 \cup R_3$ . . . . .	17
2.4	A solução do Problema de Riemann: com adsorção. . . . .	17
2.4.1	CASO D : $c_e = c_d$ . . . . .	19
2.4.2	CASO E : $c_e < c_d$ . . . . .	19
2.4.3	CASO F : $c_e > c_d$ . . . . .	22
2.5	Algoritmo e experimentos numéricos . . . . .	23
<b>3</b>	<b>A Injeção de bancos</b>	<b>30</b>
3.1	Adimensionalização e cálculos preliminares . . . . .	31
3.2	Algoritmo para a solução: . . . . .	33
3.2.1	$t < t_0$ (solução da equação de Buckley-Leverett): . . . . .	33
3.2.2	$t \geq t_0$ ( Interação entre s-onda e c-choque ) . . . . .	34
3.3	A injeção de três bancos . . . . .	43
3.4	Modelo com Adsorção - Injeção de dois bancos. . . . .	44

<b>4</b>	<b>Testes Numéricos</b>	<b>46</b>
4.1	Simulações no caso sem adsorção. . . . .	46
4.2	Simulações no caso com adsorção. . . . .	49
4.3	Simulações no caso da injeção de bancos. . . . .	50
4.4	O método numérico e o semi-analítico . . . . .	56
<b>5</b>	<b>Conclusões</b>	<b>60</b>
5.1	Trabalhos Futuros . . . . .	61
<b>6</b>	<b>Apêndice A</b>	<b>63</b>
6.1	A descrição do modelo . . . . .	63
6.2	Solução fraca . . . . .	65
6.3	Ondas simples . . . . .	65
<b>7</b>	<b>Apêndice B</b>	<b>68</b>
7.1	O Problema de Riemann - Modelo sem adsorção. . . . .	68
7.1.1	Dados.m . . . . .	69
7.1.2	Buck.M . . . . .	69
7.1.3	Isacmtd.m . . . . .	70
7.1.4	Prodisac.m . . . . .	70
7.1.5	Upwisac.m . . . . .	71
7.2	O Problema de Riemann - Modelo com adsorção. . . . .	71
7.2.1	Dados.m . . . . .	71
7.2.2	Joha.m . . . . .	71
7.2.3	Prodjoh.m . . . . .	72
7.2.4	Intstar.m . . . . .	73
7.2.5	Inter1.m . . . . .	73
7.2.6	Interk.m . . . . .	74
7.2.7	Rotst.m . . . . .	74
7.2.8	Upwjoha.m . . . . .	75
7.3	O Problema da Injeção de Bancos - Caso sem adsorção. . . . .	75
7.3.1	Dados.m . . . . .	76
7.3.2	Curvalg.m . . . . .	76
7.3.3	Curvay1.m . . . . .	76
7.3.4	Curvay2.m . . . . .	77
7.3.5	Curvay3.m . . . . .	77
7.3.6	Prodbanc.m . . . . .	78

7.3.7	Upwalg.m . . . . .	79
7.4	O Problema da Injeção de Bancos - com adsorção. . . . .	79
7.4.1	Dados.m . . . . .	80
7.4.2	Curvas.m . . . . .	80
7.4.3	Twobanc.m . . . . .	80
7.4.4	Prodads.m . . . . .	80
7.4.5	Upwads.m . . . . .	81

# 1. INTRODUÇÃO

O estudo de processos que incrementam a extração de petróleo tem merecido atenção nas últimas décadas. A essência destes processos está na alteração de propriedades da água, do óleo e do meio poroso, de modo que o deslocamento do óleo seja mais eficiente e a sua recuperação seja melhorada. Uma possibilidade importante consiste em adicionar aditivos dinamicamente ativos no processo de injeção de água em reservatórios petrolíferos.

As simulações destes processos repousam na teoria utilizada para descrever fenômenos hidrodinâmicos acompanhados de reações químicas e transferência de massa. Neste caso considera-se o fluxo bifásico, água e óleo, no qual a água contém uma substância, chamada aditivo ativo, capaz de alterar a hidrodinâmica do fluxo. O efeito hidrodinâmico do aditivo, está na sua influência nos componentes básicos que caracterizam a dinâmica do fluxo em meios porosos, isto é, nas permeabilidades, nas viscosidades e na pressão capilar. No caso unidimensional é crucial que saibamos como o aditivo influencia a função fluxo fracionário, que neste caso é função não apenas da saturação de água, mas também da concentração do aditivo. A influência na função de fluxo fracionário se deve à modificação da viscosidade na presença do aditivo.

A obtenção do modelo matemático que simula o processo de recuperação secundária de petróleo por meio de aditivos é baseada nas equações de balanço de material. Assumindo que fluido e rocha são incompressíveis, que os volumes não mudam quando o polímero é dissolvido na água, o princípio da conservação de massa de água, óleo e aditivo num escoamento bifásico unidimensional, verificamos que tais processos são modelados por meio de um sistema de equações de diferenciais parciais hiperbólico de leis de conservação.

Uma das técnicas utilizadas na recuperação do petróleo em reservatórios é a injeção de bancos de água e água com polímeros. Consiste em injetarmos água no reservatório durante um certo período de tempo e em seguida injetar a mistura de água com polímero. Neste caso dois bancos são injetados, mas de um modo geral, podemos injetar mais bancos de água, com ou sem polímero. Este processo

melhora a eficiência da recuperação em comparação com o da injeção somente de água, como mostraremos no decorrer do nosso trabalho. A injeção de bancos é uma alternativa economicamente viável, já que na prática a injeção de água com polímeros simplesmente, encarece muito o processo.

## 1.1. Revisão Bibliográfica:

Dentre os trabalhos publicados sobre o assunto cabe destacar alguns que são importantes. São os trabalhos de Isaacson (ver [ISA]), Johansen (ver [JOH]), Barenblatt (ver [BRB]) e Bedrikovetsky (ver [BED]) que tratam a parte teórica. Uma leitura interessante é o livro de Le Veque (ver [LVQ]) que faz um estudo tanto da parte numérica quanto da parte analítica da solução de problemas associados à leis de conservação. Passamos a um breve comentário sobre cada um deles.

### 1.1.1. O Trabalho de Isaacson

Neste trabalho, é apresentada uma solução global (fraca) do Problema de Riemann associado ao sistema não estritamente hiperbólico de leis conservação. Tal sistema é dado pelas equações:

$$\begin{cases} s_t + f(s, c)_x = 0 \\ (s.c)_t + (c.f(s, c))_x = 0 \end{cases} \quad (\text{I})$$

A não hiperbolicidade vem do fato que existe uma curva no espaço de estados em que velocidades características coincidem. A interpretação física da solução e o problema da entropia são também destacados neste trabalho.

### 1.1.2. O Trabalho de Johansen & Winther

Neste trabalho os autores apresentam uma solução (fraca) do problema de Riemann para um modelo que é uma generalização do proposto por Isaacson, uma vez que introduzem o efeito da adsorção do polímero pela rocha. Considerando o fator adsorção, o sistema de equações fica da forma:

$$\begin{cases} s_t + f(s, c)_x = 0 \\ (s.c + a(c))_t + (c.f(s, c))_x = 0 \end{cases} \quad (\text{II})$$

A adsorção do polímero pela rocha é modelada pelo termo  $a(c)$  que aparece no sistema acima. O modelo (II) é portanto mais realístico que o modelo (I).

Matematicamente, o efeito é que campo característico linearmente degenerado que aparece na análise dada em Isaacson (ver [ISA]), é trocado por um campo não-degenerado. As descontinuidades de contato na solução apresentada por Isaacson são trocadas por ondas de rarefação e ondas de choque.

### **1.1.3. Os Trabalhos de Barenblatt & outros e Bedrikovetsky**

Os trabalhos de Barenblatt & outros (ver [BRB]) e Bedrikovetsky (ver [BED]) tratam do método da injeção de bancos na recuperação do petróleo. O problema fica caracterizado matematicamente pelo sistema de equações hiperbólicas de leis de conservação ( I ) ou ( II ) com condições iniciais e de fronteira apropriadas. Neste trabalho as soluções são apresentadas apenas gráficamente. Elas servirão de base para o desenvolvimento do algoritmo para a solução semi-analítica que vamos apresentar.

## **1.2. Objetivos:**

Nosso objetivo é desenvolver e analisar algoritmos que possam ser usados na simulação do processo de recuperação de petróleo. As soluções do Problema de Riemann apresentadas por Isaacson ( [ISA] ) para o caso sem adsorção e por Johansen&Whinter ( [JOH] ) para o caso com adsorção são de fundamental importância para o desenvolvimento deste trabalho. Para o caso da injeção de bancos, além das descontinuidades descritas pela solução do problema de Riemann, novas descontinuidades se formam com a interação entre ondas de rarefação e de choque que aparecem na solução global do problema.

Vamos apresentar os procedimentos de cálculo utilizados na construção de uma solução semi-analítica, bem como a implementação numérica visando simulações deste processo de recuperação.

## **1.3. Organização do Trabalho:**

No capítulo 2 apresentamos um resumo das soluções globais para o problema de Riemann associados aos modelos ( I ) e ( II ) que são apresentadas nos trabalhos de Isaacson (ver [ISA]) e de Johansen (ver [JOH]).

Testes numéricos são apresentados para ilustrar a solução do problema de Riemann, através da utilização de um programa computacional desenvolvido para implementa-la.

No capítulo 3 tratamos do problema da injeção de bancos como método de recuperação do petróleo, apresentando a técnica utilizada para obtenção da solução, bem como sua implementação numérica. Trabalharemos com no máximo três bancos alternados para o caso sem adsorção e dois bancos para o caso com adsorção.

No capítulo 4 apresentamos resultados numéricos para o problema através do programa desenvolvido para simular a solução obtida, fazendo comparações entre os modelos apresentados no que diz respeito à produção de óleo e condições iniciais no reservatório. Também comparamos a título de ilustração o método semi-analítico com o método de diferenças finitas tipo upwind para mostrar as vantagens de um em relação ao outro no que diz respeito a tempo computacional.

No capítulo 5 apresentamos as conclusões do trabalho e o que ainda pode ser desenvolvido em pesquisas futuras.

No apêndice A mostramos a modelagem do problema da recuperação de petróleo através da injeção de água com polímeros.

No apêndice B fazemos a documentação dos programas numéricos utilizados para a simulação das soluções semi-analíticas desenvolvidas.

## 2. O PROBLEMA DE RIEMANN

Consideremos os sistemas de equações diferenciais parciais:

$$\begin{cases} s_t + f(s, c)_x = 0 \\ (s.c)_t + (c.f(s, c))_x = 0 \end{cases} \quad (2.1)$$

$$\begin{cases} s_t + f(s, c)_x = 0 \\ (s.c + a(c))_t + (c.f(s, c))_x = 0 \end{cases} \quad (2.2)$$

O sistemas (2.1) e (2.2) são sistemas de leis de conservação não estritamente hiperbólicos e modelam o processo de recuperação de óleo por meio da injeção de polímeros misturados a água [Apêndice A]. Este processo é conhecido também por polymer-flooding e trata-se de um método terciário de recuperação de óleo em reservatórios. A diferença entre (2.1) e (2.2) está no fato de que o sistema (2.2) leva em conta o efeito da adsorção do polímero pela rocha, ao contrário do sistema (2.1). Ambos generalizam a equação de Buckley-Leverett (ver [ISA]), que modela a injeção de água como processo de recuperação, também conhecido como water-flooding.

Consideremos as seguintes condições iniciais:

$$\begin{cases} s(x, 0) = s_d & c(x, 0) = c_d & 0 < x \\ s(x, 0) = s_e & c(x, 0) = c_e & x < 0 \end{cases} \quad (2.3)$$

Resolver o sistema (2.1) ( ou (2.2) ) associado a condições do tipo (2.3) significa resolver um problema de Riemann.

Em seu trabalho Isaacson ([ISA]) mostra a existência de uma solução analítica fraca para o problema de Riemann (2.1)-(2.3). Com hipóteses feitas sobre a função  $a(c)$  que modela o efeito da adsorção, Johansen([JOH]) mostra a existência de uma solução do problema de Riemann (2.2)-(2.3).

Nosso objetivo neste capítulo será mostrar um resumo da solução para ambos os casos e realizarmos testes numéricos utilizando o programa computacional em que implementamos estas soluções para alguns casos. Faremos inicialmente alguns comentários a respeito dos processos de recuperação citados.



## 2.1. A injeção de água

Este processo consiste da injeção de água no poço chamado de injetor que desloca o óleo para o poço chamado produtor. Tal processo é modelado pela equação de Buckley-Leverett. Com as hipóteses que os fluidos são incompressíveis e unidimensionais esta equação na forma adimensional é dada por:

$$s_t + f(s)_x = 0 \quad (2.4)$$

onde,  $s$  é saturação de água ( $0 \leq s \leq 1$ ) e  $f(s)$ , conhecida como função de fluxo fracionário representa a parcela de água no volume do fluxo em deslocamento.

A função  $f(s)$  é dada pela expressão:

$$f(s) = \frac{K_w(s)}{K_w(s) + K_o(s) \cdot \mu_w \mu_o^{-1}} \quad (2.5)$$

onde  $K_w(s)$  é a permeabilidade relativa do meio poroso em função da saturação de água  $s$ ;  $\mu_w$  mede a viscosidade da água e  $K_o(s)$ ,  $\mu_o$  medem respectivamente a permeabilidade e a viscosidade do óleo. Para efeito de cálculo vamos considerar que  $K_w$  e  $K_o$  dependem de  $s$  da seguinte forma:

$$K_w(s) = s^2 \quad K_o(s) = (1 - s)^2$$

Com estas considerações e supondo que  $\mu_w \mu_o^{-1} = cte$  mostramos na Figura 2.1 como seria uma curva típica do fluxo fracionário.

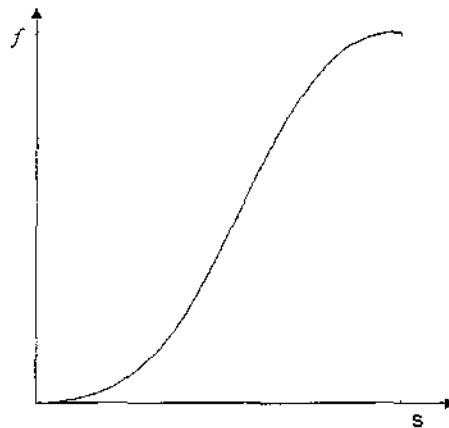


Figura 2.1 : Curva típica do fluxo fracionário

A forma da função de fluxo fracionário, exerce influência na solução da equação 2.1 ( Buckley-Leverett). Por exemplo, consideremos duas curvas do fluxo fracionário  $f_1(s)$  e  $f_2(s)$  como a da Figura 2.2.

Na Figura 2.3 mostramos como ficam os perfis da solução de saturação  $s_1(x, t)$  e  $s_2(x, t)$  correspondente à estas duas curvas, depois de um certo período de injeção de água no reservatório. Podemos dizer que a recuperação de óleo está associada à área sob o gráfico do perfil  $s(x, -)$ . Uma área maior significa que temos mais água do que óleo no reservatório e portanto a quantidade de óleo produzida é maior para a curva de área maior. A recuperação associada a  $f_1(s)$  é portanto menor que a associada a  $f_2(s)$  durante o período de injeção de água no reservatório.

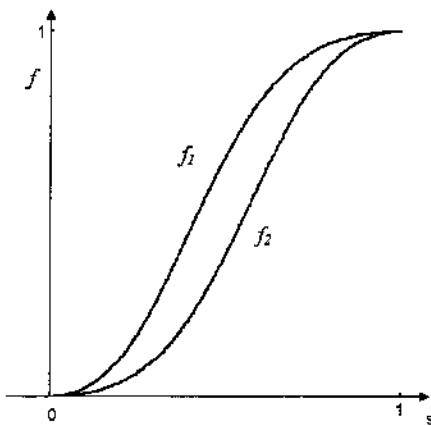


Figura 2.2

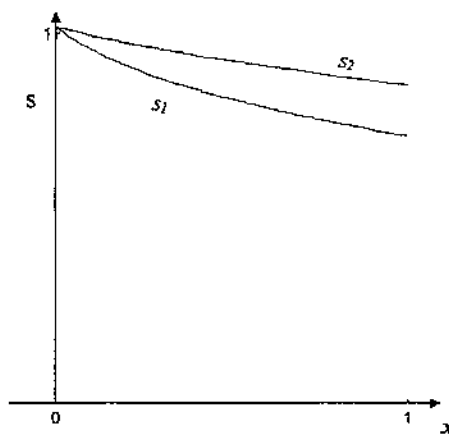


Figura 2.3

## 2.2. A injeção de polímero ( polymer-flooding ).

O método da injeção de polímeros adicionados à água em reservatórios, conhecido como polymer-flooding é uma etapa posterior ao water-flooding. Ele tem como objetivo melhorar a eficiência do processo de recuperação.

Alguns fatores responsáveis pelo aumento da eficiência da recuperação são, por exemplo, o aumento da viscosidade da água que faz com que os dois fluidos fiquem mais parecidos, ou ainda o fato de o polímero preencher alguns espaços por onde a água escoava, inibindo seu escoamento ali.

No presente modelo vamos assumir que somente o aumento da viscosidade esteja agindo. Além disso vamos supor que o polímero seja totalmente miscível na água. A equação de conservação de água neste caso é dada por:

$$s_t + f(s, c)_x = 0 \quad (2.6)$$

uma vez que o fluxo fracionário depende também da concentração de polímero  $c$  na água. Neste caso a função de fluxo fracionário é dada pela expressão:

$$f(s, c) = \frac{K_w(s)}{K_w(s) + K_o(s) \cdot \mu_w(c) \mu_o^{-1}} \quad (2.7)$$

O termo  $\mu_w(c)$  mede a viscosidade da mistura água-polímero ( este último com concentração  $c$  ) e  $\mu_o$  é a viscosidade do óleo. Os termos  $K_w, K_o$  são respectivamente as permeabilidades relativas da água e do óleo. Como  $\mu_w(c)$  é uma função crescente de  $c$ , então  $f(s, c)$  decresce quando  $c$  cresce como vemos na Figura 2.4.

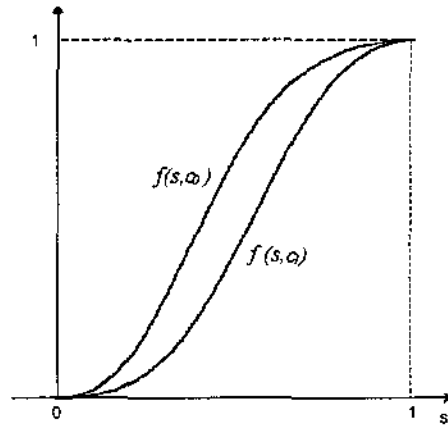


Figura 2.4 : Gráficos da função  $f(s, c)$  para dois valores de  $c$  (  $c_1 > c_0$  ).

Para efeito de cálculo consideramos a dependência de  $c$ , pela expressão (ver [ISA] ):

$$\mu_w(c) \cdot \mu_o^{-1} = (0.5 + 100 \cdot c)$$

### 2.3. A solução do Problema de Riemann: sem adsorção.

Como definimos anteriormente, resolver o problema de Riemann associado à (2.1) consiste em encontrar uma solução fraca que satisfaça as condições iniciais e de contorno:

$$\begin{cases} s(x, 0) = s_d & c(x, 0) = c_d & 0 < x \\ s(x, 0) = s_e & c(x, 0) = c_e & 0 > x \end{cases} \quad (2.8)$$

Chamamos  $u_e = (s_e, c_e)$  e  $u_d = (s_d, c_d)$  de estado à esquerda e à direita respectivamente do problema de Riemann.

Vamos descrever a solução apresentada por Isaacson ([ISA]) para o Problema de Riemann. Veremos que tal solução consiste da composição de ondas centradas simples, ondas de choque e descontinuidades de contato, dependendo da localização dos estados à direita e à esquerda no plano  $(s, c)$ . Algumas hipóteses são feitas sobre a função de fluxo fracionário. Matematicamente consideramos a função  $f(s, c)$  suave em  $I \times I$  ( $I = [0, 1]$ ) e satisfazendo as condições:

- $f(0, c) = 0$  e  $f(1, c) = 1$ ;
- $f_s(s, c) > 0$  para  $0 < s < 1$  e  $0 \leq c \leq 1$ ;
- $f_c(s, c) < 0$  para  $0 < s < 1$  e  $0 \leq c \leq 1$ ;
- Para cada  $c \in I = [0, 1]$  a função  $f(s, c)$  tem um único ponto de inflexão  $s^I(c) \in (0, 1)$  tal que  $f_{ss}(s, c) > 0$  para  $0 < s < s^I$  e  $f_{ss}(s, c) < 0$  para  $s^I < s < 1$ .

Vamos agora ao cálculo das velocidades características do sistema de equações 2.1.

Desenvolvendo formalmente as derivadas da segunda equação do sistema (2.1) temos agrupando termos:

$$c \cdot (s_t + f(s, c)_x) + s \cdot c_t + c_x f(s, c) = 0$$

A primeira parcela é eliminada utilizando a primeira equação de (2.1) e obtemos:

$$c_t + c_x \cdot \frac{f(s, c)}{s} = 0 \quad (2.9)$$

Desenvolvendo as derivadas da primeira equação do sistema (2.1) juntando com (2.9) temos

$$\begin{cases} s_t + f_s(s, c)s_x + f_c(s, c)c_x = 0 \\ c_t + c_x \cdot \frac{f(s, c)}{s} = 0 \end{cases} \quad (2.10)$$

ou ainda,

$$u_t + A(u) \cdot u_x = 0 \quad (2.11)$$

onde  $u = \begin{bmatrix} s \\ c \end{bmatrix}$  e  $A(u)$  é a matriz dada por:

$$A(u) = \begin{bmatrix} f_s(s, c) & f_c(s, c) \\ 0 & \frac{f(s, c)}{s} \end{bmatrix} \quad (2.12)$$

Os autovalores da matriz  $A(u)$  são chamados de velocidades características do sistema e são dados por:

$$\lambda^s(s, c) = f_s(s, c) \quad , \quad \lambda^c(s, c) = \frac{f(s, c)}{s} \quad (2.13)$$

Aos autovalores  $\lambda^s$  e  $\lambda^c$  estão associados os autovetores

$$e^s = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

$$e^c = \begin{pmatrix} f_c \\ \lambda^c - \lambda^s \end{pmatrix}$$

respectivamente.

O sistema não é estritamente hiperbólico já que as velocidades características coincidem sobre uma curva, que será denotada por  $T = T(s, c)$  no plano fases  $(s, c)$ . Esta curva divide o plano em duas regiões  $L$  e  $R$  que são definidas por:

$$T = \{(s, c) \mid \lambda^s(s, c) = \lambda^c(s, c)\} \quad (2.14)$$

$$L = \{(s, c) \mid \lambda^s(s, c) < \lambda^c(s, c)\} \quad (2.15)$$

$$R = \{(s, c) \mid \lambda^s(s, c) > \lambda^c(s, c)\} \quad (2.16)$$

A Figura 2.5 mostra um esboço da curva de transição  $T$  e das regiões  $L$  e  $R$

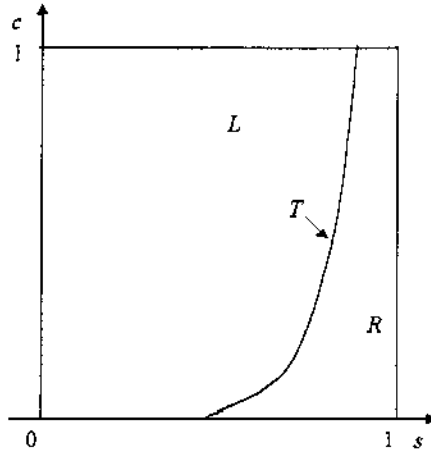


Figura 2.5 : Curva de Coincidência  $T$  e as Regiões  $L$  e  $R$

Para descrever composição de ondas na construção da solução final vamos utilizar a seguinte notação:

$u_e \xrightarrow{c} u_d$ , significa que o estado à esquerda  $u_e$  está conectado ao estado à direita  $u_d$  por uma  $c$ -onda no plano das fases.

$u_e \xrightarrow{s} u_d$  significa que o estado à esquerda  $u_e$  está conectado ao estado à direita  $u_d$  por uma  $s$ -onda que é a solução da equação de Buckley-Leverett.

$u_e \xrightarrow{a} u_i \xrightarrow{b} u_d$  significa uma composição de ondas, onde  $a$  ou  $b$  podem ser  $s$  ou  $c$  e  $u_i$  é chamado de estado intermediário.

Dizemos que a composição é compatível se a velocidade final da onda  $a$  é menor que a velocidade inicial da onda  $b$ .

Dados dois estados para o problema de Riemann,  $u_e$  e  $u_d$ , a solução é dividida em três casos A, B e C, que passamos a descrever.

### 2.3.1. CASO A : $c_d = c_e$ .

Este caso é o mais simples, pois a solução é uma  $s$ -onda dada pela resolução da equação de Buckley-Leverett e será representada por:

$$u_e \xrightarrow{s} u_d \quad (\text{A})$$

Lembremos que neste caso a solução pode ser um  $s$ -choque, uma  $s$ -rarefação ou ainda uma composição destas duas. Como ilustração a Figura 2.6 mostra perfis da saturação da água para três valores de tempo  $t$ , quando consideramos a saturação à esquerda  $s_e = 1$  e a saturação à direita  $s_d = 0$  e  $c_e = c_d$ .

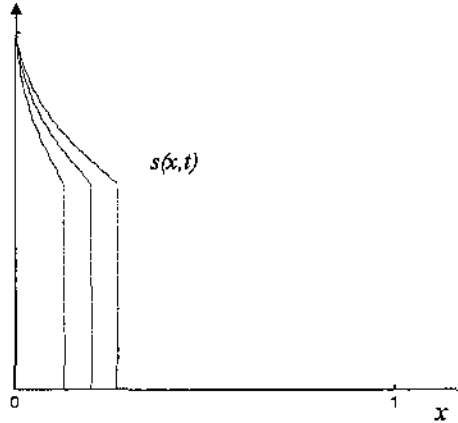


Figura 2.6 : Perfis da solução para três valores de  $t$

Antes de passarmos ao próximo caso vamos apresentar a construção de alguns subconjuntos de  $I \times I$  ( $I = [0, 1]$ ) que aparecem na solução.

Considere  $u_e \in L$  e a curva integral  $\gamma(t) = (\gamma_1(t), \gamma_2(t))$  associada ao campo  $e^C$  (a partir de agora será chamada de  $c$ -curva integral) que passa por  $u_e$  e intersecta a curva de transição  $T$  em um ponto  $(s^T, c^T)$ . Utilizando esta curva integral vamos definir três regiões que dependem do estado  $u_e = (s_e, c_e)$ :

- $L_1 = L_1(u_e)$  é a região dos pontos  $(s, c)$  tais que  $c_e \leq c \leq c^T$  e  $0 \leq s \leq \gamma_1^R(t)$ . ( $\gamma^R(t)$  é a parte da curva integral contida na região  $R$ ).
- $L_2 = L_2(u_e)$  é a região dos pontos  $(s, c)$  tais  $(s, c) \in R$  com  $c_e \leq c \leq 1$  e  $\gamma_1^R(t) < s \leq 1$
- $L_3 = L_3(u_e)$  é a região dos pontos  $(s, c)$  tais  $(s, c) \in L$  com  $c^T \leq c \leq 1$ .

A Figura 2.7 mostra um esboço das regiões  $L_1, L_2$  e  $L_3$  definidas acima.

Se  $u_e \in R$ , consideramos o estado  $u^T = (s^T(c_e), c_e) \in T$  e a  $c$ -curva integral  $\gamma(t) = (\gamma_1(t), \gamma_2(t))$  associada ao campo  $e^C$ . Definimos então as regiões  $R_1 = R_1(u_e)$ ,  $R_2 = R_2(u_e)$  e  $R_3 = R_3(u_e)$  da seguinte maneira:

- $R_1 = R_1(u_e)$  é a região dos pontos  $(s, c)$  do plano das fases  $s \times c$  tais que  $0 \leq c \leq c_e$  e  $0 \leq s \leq \gamma_1^R(t)$ .
- $R_2 = R_2(u_e)$  é a região dos pontos  $(s, c)$  do plano das fases tais  $(s, c) \in R$  com  $0 \leq c \leq 1$  e  $\gamma_1^R(t) < s \leq 1$

- $R_3 = R_3(u_e)$  é a região dos pontos  $(s, c)$  do plano das fases tais  $(s, c) \in L$  com  $c_e < c \leq 1$ .

A Figura 2.8 mostra um esboço destas regiões.

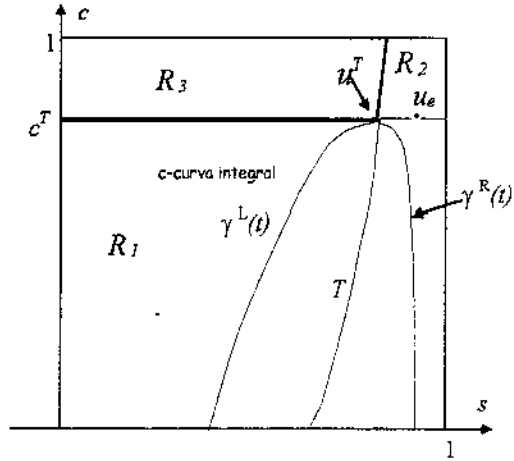
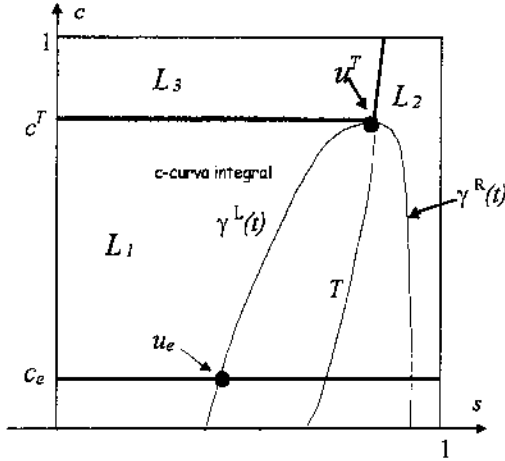


Figura 2.7 : As regiões  $L_1, L_2$  e  $L_3$ .

Figura 2.8 : As regiões  $R_1, R_2$  e  $R_3$ .

### 2.3.2. CASO B: $u_e \in L$ e $u_d \in L_1 \cup L_2 \cup L_3$

Teremos então três composições de ondas compatíveis formando a solução. Elas dependem da localização de  $u_d$  em  $L_1, L_2$  ou  $L_3$  (Figura 2.7)

Se  $u_d \in L_1$ . A solução é dada pela composição de ondas compatíveis:

$$u_e \xrightarrow{c} u_1 \xrightarrow{s} u_d \quad (\text{B.i})$$

onde  $u_1$  é dado pela interseção da curva integral do campo  $e^C$  ( neste caso as curvas integrais associadas são descontinuidades de contato ) que passa por  $u_e$  com a reta  $c = c_d$ .

Se  $u_d \in L_2$ , temos a seguinte composição:

$$u_e \xrightarrow{s} u_1 \xrightarrow{c} u_d \quad (\text{B.ii})$$

onde o estado intermediário  $u_1$  é calculado como a interseção da c-curva integral que passa por  $u_d$  com a reta  $c = c_e$

Se  $u_d \in L_3$ , temos a seguinte composição:



$$u_e \xrightarrow{s} u_1 \xrightarrow{c} u_2 \xrightarrow{s} u_d \quad (\text{B.iii})$$

onde  $u_1$  é calculado como no caso anterior e  $u_2$  é a interseção da reta  $c = c_d$  com a curva de transição  $T$ .

### 2.3.3. CASO C : $u_e \in R$ e $u_d \in R_1 \cup R_2 \cup R_3$

Suponhamos que  $u_d \in R_1$ , então teremos a seguinte composição de ondas:

$$u_e \xrightarrow{s} u_1 \xrightarrow{c} u_2 \xrightarrow{s} u_d \quad (\text{C.i})$$

onde  $u_1$  é a interseção da reta  $c = c_e$  com a curva de transição  $T$  e  $u_2$  é interseção da c-curva curva integral que passa por  $u_1$  com a reta  $c = c_d$ .

Se  $u_d \in R_2$ , então teremos a seguinte composição de ondas:

$$u_e \xrightarrow{s} u_1 \xrightarrow{c} u_d \quad (\text{C.ii})$$

onde  $u_1$  é a interseção da c-curva curva integral que passa por  $u_d$  com a reta  $c = c_e$ .

Suponhamos que  $u_d \in R_3$ , então temos a seguinte composição de ondas:

$$u_e \xrightarrow{s} u_1 \xrightarrow{c} u_2 \xrightarrow{s} u_d \quad (\text{C.iii})$$

onde  $u_2$  é a interseção da reta  $c = c_d$  com a curva de transição  $T$  e  $u_1$  é interseção da c-curva curva integral que passa por  $u_2$  com a reta  $c = c_e$ .

Do exposto vemos que o caso sem adsorção admite sete tipos de solução que dependem da localização dos estados à direita e à esquerda no plano das fases. A seguir passamos a examinar a solução para o problema com adsorção.

## 2.4. A solução do Problema de Riemann: com adsorção.

Como vimos no início do capítulo o Problema de Riemann associado ao modelo com adsorção é dado pelo sistema de leis de conservação hiperbólicas (2.2) com condições iniciais e de contorno da forma (2.3).

Este foi o modelo estudado por Johansen (ver [JOH]), sendo o efeito da adsorção, representada pela função  $a(c)$ , que deve satisfazer algumas hipóteses adicionais enunciadas no referido artigo.

Como no caso anterior a solução do problema de Riemann consistirá da composição de uma sequência compatível de s-ondas e c-ondas. Porém observamos

que neste caso as  $c$ -ondas podem ser  $c$ -rarefações, o que não acontece no modelo sem adsorção.

O sistema (2.2) pode ser reescrito também na forma matricial, ou forma não conservativa:

$$u_t + A(u).u_x = 0 \quad (2.17)$$

onde  $u = \begin{bmatrix} s \\ c \end{bmatrix}$  e  $h(c) = a'(c)$  e

$$A(u) = \begin{bmatrix} f_s(s, c) & f_c(s, c) \\ 0 & \frac{f(s, c)}{s + h(c)} \end{bmatrix} \quad (2.18)$$

Os auto-valores da matriz  $A$  são dados neste caso por :

$$\lambda^s(s, c) = f_s(s, c) \quad , \quad \lambda^c(s, c) = \frac{f(s, c)}{s + h(c)} \quad (2.19)$$

com autovetores associados dados por:

$$e^s = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

$$e^c = \begin{pmatrix} f_c \\ \lambda^c - \lambda^s \end{pmatrix}$$

Além dos três conjuntos  $L$ ,  $R$  e  $T$  (este sistema também não é estritamente hiperbólico), definidos de maneira análoga aqueles definidos em (2.14), (2.15) e (2.16), veremos que outros três conjuntos desempenharão também papel importante na caracterização da solução do problema de Riemann.

Vamos apresentar um resumo da solução analítica do problema de Riemann, mostrando todas as possíveis seqüências compatíveis de composições de ondas que dependem dos valores das concentrações e saturações à esquerda e à direita, do problema de Riemann.

Temos também três casos a considerar:  $D$ ,  $E$  ou  $F$ , que passamos a descrever.

### 2.4.1. CASO D : $c_e = c_d$

Supondo  $c_e = c_d$ , a solução neste é a mesma dada em (A), ou seja:

$$u_e \xrightarrow{s} u_d \quad (D)$$

Antes de passarmos ao caso E vejamos a seguinte definição:

**Definição:** Dado  $u = (s, c) \in R \cup T$ , definimos  $s^k(u)$ , como sendo o único valor  $s = s^k$ , com  $u^k = (s^k, c) \in L \cup T$  ( ver Figura 2.9) tal que:

$$\lambda^c(s^k, c) = \lambda^c(s, c) \quad (2.20)$$

Analogamente se  $u = (s, c) \in L$ ,  $s^k = s^k(u)$  é tal que  $(s^k, c) \in R$  e que (2.20) é verdadeiro ou se tal  $s^k$  não existe, tomamos  $s^k$  como sendo uma constante suficientemente grande.

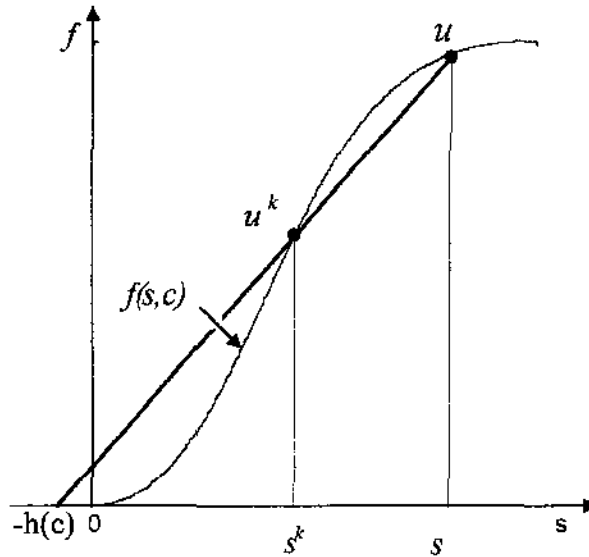


Figura 2.9 : Cálculo de  $u^k = (s^k(u), c)$

### 2.4.2. CASO E : $c_e < c_d$

Neste caso a solução dependerá das localizações de  $u_e$  e  $u_d$

**E.1)  $u_e \in R \cup T$**

A solução depende da localização do ponto  $u_d$

1. Se  $u_d \in R \cup T$ , então tomamos  $u_1 = (s_1, c_e) \in R$  determinado pela interseção da  $c$ -curva integral que passa por  $u_d$  com a reta  $c = c_e$ . A solução única neste caso é dada pela composição:

$$u_e \xrightarrow{s} u_1 \xrightarrow{c} u_d \quad (\text{E.1.i})$$

2. Se  $u_d \in L$ , verificamos se  $s^T(c_d)$  existe. Se não existir tomamos a composição:

$$u_e \xrightarrow{s} (1, c_e) \xrightarrow{c} (1, c_d) \xrightarrow{s} u_d \quad (\text{E.1.ii})$$

3. Se  $s^T(c_d)$  existe tomamos a composição:

$$u_e \longrightarrow u_T \xrightarrow{s} u_d \quad (\text{E.1.iii})$$

onde  $u^T = (s^T(c_d), c_d)$  e a onda  $u_e \longrightarrow u_T$  é do tipo (E.1.i).

**E.2)  $u_d \in L$**

Aqui também a solução é construída a partir da localização do estado  $u_d$ . Denotaremos então por  $\Gamma_d \in L \cup T$ , a  $c$ -curva integral que começa em  $u_e \in L$  e termina em  $u^T = (s^T, c^T)$  interseção da  $c$ -curva integral com a curva de transição  $T$ . Caso não exista esta interseção, tomamos  $u^T = (s^T, 1)$  que seria o ponto onde a curva  $\Gamma_d$  intercepta a reta  $c = 1$  no espaço  $I \times I$ .

Denotemos por  $\Gamma_k \in R \cup T$  (Ver Fig. 2.10) como sendo a curva crítica associada à  $\Gamma_d$  definida por:

$$\Gamma_k = \{(s^k, c) \in R \cup T, s^k = s^k(s, c), \forall (s, c) \in \Gamma_d\} \quad (2.21)$$

Assim temos três regiões  $\Omega_1 = \Omega_1(u_e)$ ,  $\Omega_2 = \Omega_2(u_e)$  e  $\Omega_3 = \Omega_3(u_e)$  dadas por:

- $\Omega_1 = \Omega_1(u_e)$  conjunto dos pontos  $(s, c) \in I \times I$  delimitados pelas retas  $s = 0, c = c_e, c = c^T$  e pela curva  $\Gamma_k$ .
- $\Omega_2 = \Omega_2(u_e)$  conjunto dos pontos  $(s, c) \in R$  delimitados pelas retas  $s = 1, c = c_e, c = 1$  e a curva  $\Gamma_k$ .

- $\Omega_3 = \Omega_3(u_e)$  conjunto dos pontos  $(s, c) \in L$  delimitados pelas retas  $s = 0, c = c^T, c = 1$ .

Um esboço destas regiões está ilustrado na Figura 2.10

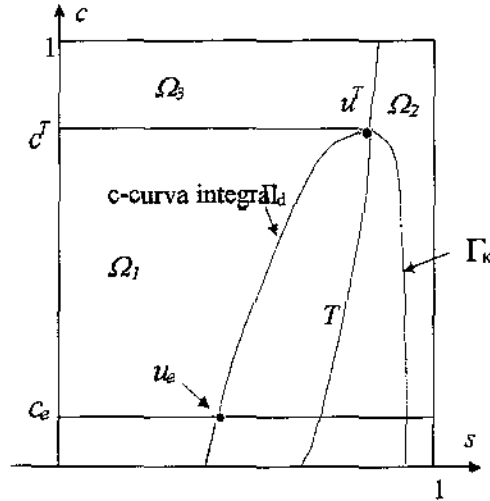


Figura 2.10 : As regiões  $\Omega_1, \Omega_2$  e  $\Omega_3$

A solução é dada em função da localização de  $u_d$  nas três regiões acima que passamos a descrever:

1. Suponha que  $u_d \in \Omega_1$ . Neste caso tomamos  $u_1 = (s_1, c_d) \in \Gamma_d$  como sendo a interseção de  $\Gamma_d$  com a reta  $c = c_d$  e consideramos a seguinte composição compatível:

$$u_e \xrightarrow{c} u_1 \xrightarrow{s} u_d \quad (\text{E.2.i})$$

2. Suponha agora que  $u_d \in \Omega_2$ . Considere o estado  $u_2 = (s_2, c_2)$  dado pela interseção da c-rarefação que passa pôr  $u_d$ , com a curva  $\Gamma_d \cup \{c = c_e\}$ . Temos aqui duas hipóteses a considerar. Se  $c_2 = c_e$ , temos a seguinte composição compatível:

$$u_e \xrightarrow{s} u_2 \xrightarrow{c} u_d \quad (\text{E.2.ii})$$

Caso contrário, se  $c_2 > c_e$ , então tomamos  $u_1 = (s_1, c_2) \in \Gamma_d$  tal que  $s^k(u_2) = s_1$  e a composição compatível é dada por:

$$u_e \xrightarrow{c} u_1 \xrightarrow{s} u_2 \xrightarrow{c} u_d \quad (\text{E.2.iii})$$

1. Seja  $u_d \in \Omega_3$ . Se  $s^T(c_d)$  existe, a composição:

$$u_e \longrightarrow u_T \xrightarrow{s} u_d$$

onde  $u_T = (s^T(c_d), c_d)$  e a onda  $u_e \longrightarrow u_T$  é da forma (E.2.iii), é compatível.

Pôr outro lado se  $s^T(c_d)$  não existe, a composição compatível é dada por:

$$u_e \xrightarrow{s} (1, c_T) \xrightarrow{c} (1, c_d) \xrightarrow{s} u_d \quad (\text{E.2.v})$$

Verifica-se também que estas composições compatíveis, são únicas.

### 2.4.3. CASO F : $c_e > c_d$

Aqui dois caso são considerados:  $\lambda^s(u_e) \geq \delta(u_e)$  e  $\lambda^s(u_e) < \delta(u_e)$ .

Observamos que as c-ondas neste caso serão sempre ondas de choque, de acordo com a análise apresentada por [JOH] e com velocidade de choque  $\delta(u)$  dada por:

$$\delta(u) = \frac{f(s, c)}{s + h_L(c_d)} \quad (2.22)$$

Dado  $u = (s, c)$  com  $\lambda^s(u) \leq \delta(u)$ , definimos  $s^k(u)$  como sendo o único estado  $u_k = (s^k(u), c)$  tal que  $\lambda^s(s^k, c) > \delta(u)$  e

$$\delta(u_k) = \delta(u) \quad (2.23)$$

Se  $\lambda^s(u) > \delta(u)$  então  $s^k = s^k(u)$ , caso exista é o único estado tal que (2.23) é verdadeiro; caso tal  $s^k$  não exista então tomamos  $s^k$  um constante suficientemente grande.

#### F.1) $\lambda^s(u_e) > \delta(u_e)$

Consideremos  $u_1 = (s_1, c_d)$  o único estado tal que  $\delta(u_1) = \delta(u_e) < \lambda^s(u_1)$ . Neste caso três possíveis soluções aparecem conforme  $s_d > s^k(u_1)$ ,  $s_d < s^k(u_1)$  ou  $s_d = s^k(u_1)$ .

1. Se  $0 \leq s_d < s^k(u_1)$  a solução é dada por:

$$u_e \xrightarrow{c} u_1 \xrightarrow{s} u_d \quad (\text{F.1.i})$$

2. No caso  $s_d > s^k(u_1)$ , seja  $u_2 = (s_2, c_e)$  o estado intermediário onde vale  $\delta(u_d) = \delta(u_2) = \lambda^s(u_2)$  e a solução é dada pela composição:

$$u_e \xrightarrow{s} u_2 \xrightarrow{c} u_d \quad (\text{F.1.ii})$$

3. Se  $s_d = s^k(u_1)$  temos um choque dado pela composição:

$$u_e \xrightarrow{c} u_d \quad (\text{F.1.iii})$$

com velocidade  $\delta(u_d) = \delta(u_e) = \delta$ .

## F.2 ) $\lambda^s(u_e) < \delta(u_e)$

Seja  $u^*(s^*, c_e)$  o único estado, tal que  $\lambda^s(u^*) = \delta(u^*)$  e seja  $u_1 = (s_1, c_d)$  determinado por  $\delta(u_1) = \delta(u^*) < \lambda^s(u_1)$  e seja  $s^k = s^k(u_1)$ . Aqui as soluções possíveis dependerão do fato de  $s_d \geq s^k$  ou  $s_d < s^k$ .

1. Se  $s_d \geq s^k$ , tome  $u_2 = (s_2, c_e)$  o único estado tal que  $\lambda^s(u_2) \leq \delta(u_2) = \delta(u_d)$ , e então temos a composição compatível única dada por:

$$u_e \xrightarrow{s} u_2 \xrightarrow{c} u_d \quad (\text{F.2.i})$$

2. Se  $s_d < s^k$  a solução é dada pela composição:

$$u_e \xrightarrow{s} u^* \xrightarrow{c} u_1 \xrightarrow{s} u_d \quad (\text{F.2.ii})$$

Sendo assim, apresentamos todas as 14 composições de ondas compatíveis possíveis que representam as soluções do Problema de Riemann. No artigo referente a este trabalho, os autores mostram que todas estas soluções são fisicamente corretas, através da demonstração de alguns teoremas e lemas, além de detalhar a geometria envolvida no problema.

## 2.5. Algoritmo e experimentos numéricos

Baseado nas seções anteriores, elaboramos algoritmos para implementar numericamente a solução analítica do problema de Riemann para o modelo sem adsorção [ISA] e o modelo com adsorção [JOH] para o caso em que  $c_e > c_d$ . O algoritmo foi

desenvolvido dentro do ambiente MATLAB que possui linguagem própria. Exploramos a potencialidade deste pacote, nos vários passos executados no cálculo da solução.

Lembramos que nos experimentos numéricos vamos utilizar as expressões algébricas citadas nos parágrafos anteriores e tomar a função:

$$f(s, c) = \frac{s^2}{s^2 + (0.5 + 100.c).(1 - s)^2} \quad (2.24)$$

para modelar a função de fluxo fracionário e

$$a(c) = \frac{(0.2).c}{1 + 100.c} \quad (2.25)$$

para modelar o efeito da adsorção.

A seguir simulamos alguns experimentos com o objetivo de ilustrar soluções associadas a estados, à esquerda e à direita, pré-estabelecidos. Os gráficos apresentados resultam dos cálculos realizados pelo programa computacional que implementamos.

### Experimento 1:

Vamos supor que estamos injetando água no reservatório contendo 2% de polímero. Isto corresponde a tomar  $u_e = (1, 0.02)$ . Admite-se que inicialmente não exista água e nem polímero no reservatório, o que corresponde a tomar  $u_d = (0, 0)$ . O perfil de saturação de água depois de termos injetado  $\frac{1}{3}$  do volume poroso ( $t \cong 0.33$ ) é apresentado na Figura 2.11. Tal gráfico corresponde a uma solução do tipo (F.2.ii) para o caso com adsorção.

No gráfico plotamos o perfil da concentração multiplicada por 5 para uma melhor visualização do mesmo. Observamos que a solução apresenta duas descontinuidades na saturação  $s(x, t)$  e uma descontinuidade na concentração  $c(x, t)$ .

Os dados para os seguintes experimentos correspondem a situações de pouco interesse prático. Neles a saturação de injeção é menor que 1 e admite-se que a saturação inicial no reservatório é grande. Apresentamos estes exemplos como ilustrações de soluções analíticas do problema de Riemann. Observamos que eles podem corresponder a outras situações práticas que não sejam injeção de água com aditivos na recuperação terciária do petróleo.



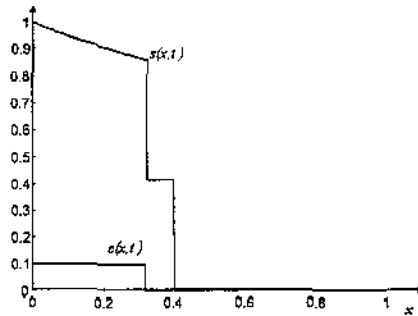


Figura 2.11 : Perfil da saturação e da concentração

### Experimento 2:

Aqui vamos considerar o caso com adsorção para um poço com saturação de água dada por,  $s_d = 0,7$  e que não exista polímero no reservatório. Injetamos então uma mistura de água, polímero e gás por exemplo, que deverá ter uma saturação de água da ordem de 30% e uma concentração de polímero da ordem de 1%. O perfil de saturação de água depois de termos injetado metade do volume poroso é apresentado na Figura 2.12.

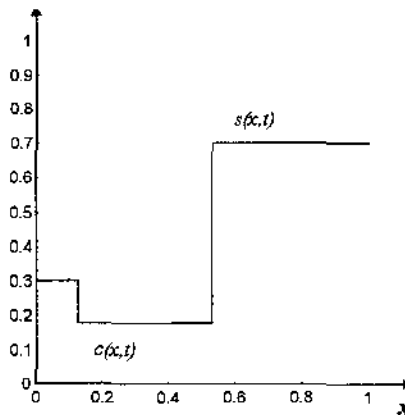


Figura 2.12 : Perfil da saturação e da concentração

### Experimento 3:

Consideremos a injeção de uma mistura com saturação de água da ordem 98% com uma concentração de polímero em torno de 1% e que o reservatório já contenha 90% de água e que não estamos considerando o efeito da adsorção do polímero pela rocha.

O gráfico da Figura 2.13 mostra os perfis da saturação e da concentração.

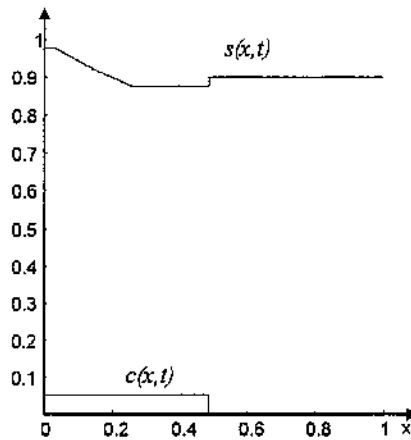


Figura 2.13 : Perfil da saturação e da concentração

### Experimento 4 :

Consideremos o caso sem adsorção onde estamos injetando uma mistura com uma saturação de água da ordem de 50% sem polímero em um reservatório com saturação de água da ordem de 20% e com 1% de concentração de polímero. A solução neste caso dada pela composição:

$$u_e \xrightarrow{s} u_1 \xrightarrow{c} u_2 \xrightarrow{s} u_d$$

A Figura 2.14 mostra os perfis de saturação de água  $s(x, t)$  e da concentração de polímero  $c(x, t)$  para este caso.

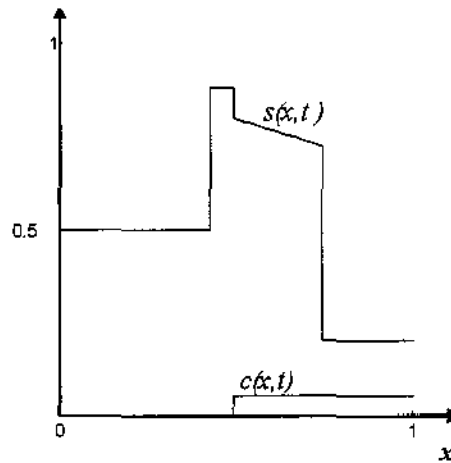


Figura 2.14 : Perfil da saturação e da concentração

No gráfico observamos que a concentração de polímero está multiplicada por 5 para uma melhor visualização.

### Experimento 5 :

Consideremos uma saturação de injeção de água da ordem de 45% sem polímero em reservatório com saturação da ordem de 79% com 1% de concentração de polímero. Os perfis estão mostrados na Figura 2.15.

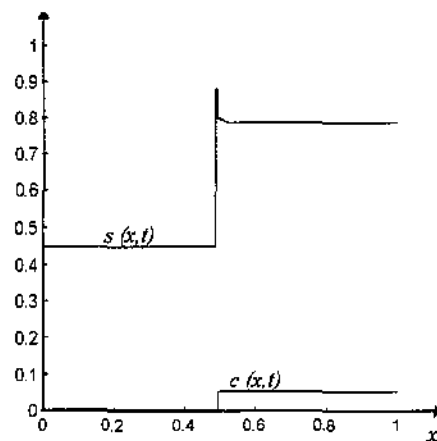


Figura 2.15 : Perfil da saturação e da concentração

### Experimento 6 :

Neste experimento fazemos a comparação entre a solução analítica do problema de Riemann e a solução dada por um esquema de diferença finitas de primeira ordem tipo upwind conforme Tveito [TVT].

Observamos que o método numérico não representa muito bem a solução em alguns casos. Considerando os dados apresentados no experimento 5, calculamos a solução numérica para dois valores de  $\Delta t$ . Neste caso para  $\Delta t$  da ordem de  $10^{-4}$  temos um tempo computacional muito grande para o cálculo da solução numérica e esta não captura o choque que aparece na solução. Este fato está ilustrado na Figura 2.16 que apresenta os gráficos dos perfis de saturação  $s(x, t)$  para os casos numérico e o analítico.

Os testes mostram ainda que as soluções obtidas pelo esquema numérico convergem para a solução semi analítica. Isto fica evidenciado na Figura 2.17, onde comparamos o caminho da solução numérica ( no plano das fases  $s \times c$  ) para alguns valores de  $\Delta t$ .

Lembramos que na figura a solução correspondente à concentração esta multiplicada por 50 para uma melhor visualização.

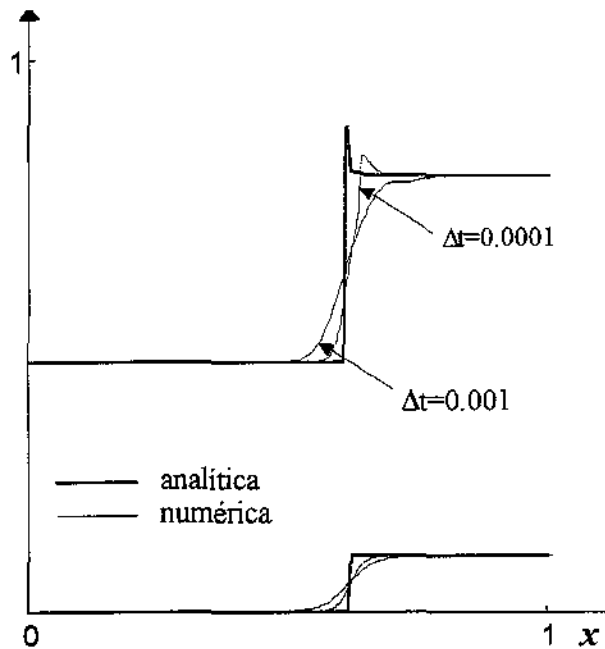


Figura 2.16

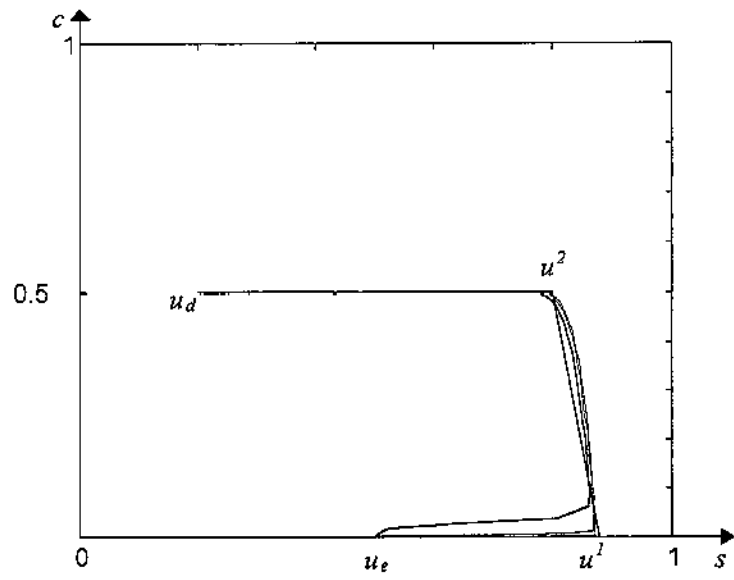


Figura 2.17

### 3. A INJEÇÃO DE BANCOS

Uma das técnicas utilizadas na recuperação do petróleo em reservatórios é a injeção alternada de bancos de água e água com polímeros. Consiste em injetarmos água no reservatório durante um certo período de tempo e em seguida injetar a mistura de água com polímero. Neste caso dois bancos são injetados, mas de um modo geral, podemos injetar mais bancos alternando bancos de água com ou sem polímero. Este processo melhora a eficiência da recuperação em comparação com o da injeção somente de água, como mostraremos no decorrer do nosso trabalho. A injeção de bancos alternados é uma alternativa economicamente viável, já que na prática a injeção de água com polímeros simplesmente, encarece muito o processo.

Nosso objetivo é descrever e analisar algoritmos que possam ser usados na simulação deste processo para os modelos com ou sem adsorção.

Matematicamente o processo de injeção alternada (por exemplo três bancos), fica caracterizado pelo sistema de equações:

$$\begin{cases} s_t + f(s, c)_x = 0 \\ (s.c)_t + (c.f(s, c))_x = 0 \end{cases} \quad (3.1)$$

para o caso sem adsorção e

$$\begin{cases} s_t + f(s, c)_x = 0 \\ (s.c + a(c))_t + (c.f(s, c))_x = 0 \end{cases} \quad (3.2)$$

com adsorção associado a condições iniciais (c.i.) e de contorno (c.c.) do tipo:

$$\begin{cases} c.i. \{ s(x, 0) = s_0; c(x, 0) = c_0 & 0 \leq x \leq 1 \\ \left\{ \begin{array}{l} s(0, t) = s_1; c(0, t) = c_1 & 0 \leq t < t_0 \\ s(0, t) = s_2; c(0, t) = c_2 & t_0 \leq t < t_1 \\ s(0, t) = s_3; c(0, t) = c_3 & t_1 \leq t \end{array} \right. \end{cases} \quad (3.3)$$

Resolver o problema (3.1)-(3.3) ou (3.2)-(3.3) significa neste caso resolver um Problema de Goursatt, já que tomamos as condições iniciais sobre a característica  $x = 0$ .

Para a solução do problema utilizaremos as soluções do Problema de Riemann apresentadas no capítulo anterior para os casos com adsorção ([JOH]) e sem adsorção ([ISA]). Para o caso da injeção de bancos alternados, além das descontinuidades descritas anteriormente, novas descontinuidades se formam com a interação entre as s-ondas e as c-ondas que aparecem na solução global do problema.

Neste capítulo vamos apresentar os procedimentos de cálculo utilizados na construção de uma solução semi-analítica, bem como a implementação numérica visando simulações deste processo de recuperação.

### 3.1. Adimensionalização e cálculos preliminares

Nas aplicações práticas vimos que a saturação de água varia entre uma saturação mínima  $s_{wi}$  e uma saturação máxima dada pela diferença  $1 - s_{or}$ , onde  $s_{or}$  é a saturação de óleo residual. Para uma maior facilidade de nossos cálculos é conveniente uma mudança de variável que faça a saturação  $s(x, t)$  variar no intervalo unitário  $I = [0, 1]$ .

A primeira equação dos sistemas, (3.1) e (3.2) modela a conservação da água no fluxo bifásico e é conhecida como equação de Buckley-Leverett:

$$\frac{\partial s_w}{\partial t_D} + \frac{\partial f_1(s_w, c)}{\partial x_D} = 0, s_{wi} \leq s_w \leq 1 - s_{or} \quad (3.4)$$

onde as variáveis independentes foram adimensionalizadas por:

$$x_D = \frac{x}{L} \quad e \quad t_D = \frac{U.t}{\phi L} \quad (3.5)$$

Para simplificar anotação vamos omitir  $D$  na equação (3.4) isto é, daqui para frente  $x$  e  $t$  representam as variáveis adimensionalizadas.

Podemos verificar facilmente que a mudança de variáveis  $s_w = s_n \cdot s + s_{wi}$  com  $s_n = 1 - s_{wi} - s_{or}$  transforma o intervalo  $s_{wi} \leq s_w \leq 1 - s_{or}$  no intervalo  $0 \leq s \leq 1$ . A a variável  $s$  será chamada de saturação normalizada.

Observando que:

$$\frac{\partial s_w}{\partial t} = s_n \frac{\partial s}{\partial t}$$

a equação da conservação da água (3.4) fica reescrita como:

$$s_n \frac{\partial s}{\partial t} + \frac{\partial f(s, c)}{\partial x} = 0 \quad (3.6)$$

onde  $0 \leq s \leq 1$  e  $f(s, c) = f_1(s_w(s), c)$ .

Com as mesmas mudanças de variáveis sendo aplicadas na equação da conservação do polímero para o sistema (3.1) ou (3.2), esta fica reescrita da forma:

$$\left(s + \frac{s_{wi}}{s_n}\right) s_n \frac{\partial c}{\partial t} + f(s, c) \cdot \frac{\partial c}{\partial x} = 0 \quad (3.7)$$

Por conveniência vamos tomar uma nova mudança de variável em  $t$ , dada por:

$$t = \frac{t_{antigo}}{s_n}$$

Assim ficamos com a seguinte equação:

$$\left(s + \frac{s_{wi}}{s_n}\right) \frac{\partial c}{\partial t} + f(s, c) \cdot \frac{\partial c}{\partial x} = 0 \quad (3.8)$$

O sistema (3.1) adimensionalizado fica escrito então na seguinte forma matricial:

$$\begin{bmatrix} \frac{\partial s}{\partial t} \\ \frac{\partial c}{\partial t} \\ \frac{\partial c}{\partial x} \end{bmatrix} + \begin{bmatrix} \frac{\partial f}{\partial s} & \frac{\partial f}{\partial c} \\ 0 & \frac{f}{s + \frac{s_{wi}}{s_n}} \end{bmatrix} \begin{bmatrix} \frac{\partial s}{\partial x} \\ \frac{\partial c}{\partial x} \\ \frac{\partial c}{\partial x} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}. \quad (3.9)$$

Os autovalores deste sistema são:

$$\lambda^s(s, c) = f_s(s, c) \quad , \quad \lambda^c(s, c) = \frac{f(s, c)}{s + \frac{s_{wi}}{s_n}} \quad (3.10)$$

Analogamente aplicamos as mudanças de variável no sistema (3.2) e obtemos:

$$\begin{bmatrix} \frac{\partial s}{\partial t} \\ \frac{\partial c}{\partial t} \\ \frac{\partial c}{\partial x} \end{bmatrix} + \begin{bmatrix} \frac{\partial f}{\partial s} & \frac{\partial f}{\partial c} \\ 0 & \frac{f}{s + h^*(c)} \end{bmatrix} \begin{bmatrix} \frac{\partial s}{\partial x} \\ \frac{\partial c}{\partial x} \\ \frac{\partial c}{\partial x} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \quad (3.11)$$

onde

$$h^*(c) = \frac{s_{wi} + a'(c)}{s_n}$$

Neste caso os autovalores são:

$$\lambda^s(s, c) = f_s(s, c) \quad , \quad \lambda^c(s, c) = \frac{f(s, c)}{s + h^*(c)} \quad (3.12)$$



### 3.2. Algoritmo para a solução:

A maior dificuldade em se obter uma solução para o caso da injeção de bancos é como tratar as novas descontinuidades que surgem na solução global. No trabalho de Isaacson [ISA] são tratadas as descontinuidades inerentes a solução do problema de Riemann associados a leis de conservação não convexa, (equação de Buckley-Leverett ) devido ao formato em  $S$  da função de fluxo fracionário.

A proposta é desenvolver uma sistemática para o cálculo da solução semi analítica que estará fundamentado nas soluções básicas do problema de Riemann dadas por Isaacson. Na realidade a estratégia é trabalhar com problemas de Riemann locais.

Para um melhor entendimento do procedimento consideremos inicialmente o modelo com injeção de dois bancos sendo o primeiro de água seguido de outro de água com polímero.

Vamos supor que um reservatório inicialmente não contenha água e nem polímero. Durante um período de tempo,  $t = t_o$ , injetamos água sem polímero e a partir daí passamos a injetar a mistura de água e polímero com concentração  $c_{inj2} \geq 0$ .

As condições iniciais (c.i.) e de contorno (c.c.) neste caso são:

$$\left\{ \begin{array}{l} c.i. \{ s(x, 0) = s_o, c(x, 0) = c_o, 0 \leq x \leq 1 \\ c.c. \left\{ \begin{array}{l} s(0, t) = 1, c(0, t) = 0, 0 \leq t \leq t_o \\ s(0, t) = 1, c(0, t) = c_{inj2}, t_o \leq t \end{array} \right. \end{array} \right. \quad (3.13)$$

Vamos fazer uma análise da solução em duas etapas: a injeção do primeiro banco ( $t < t_o$ ) e em seguida a injeção do segundo banco ( $t \geq t_o$ ).

#### 3.2.1. $t < t_o$ (solução da equação de Buckley-Leverett):

Para  $t < t_o$  a solução está bem caracterizada, pois trata-se da solução da equação de Buckley-Leverett e portanto consiste da composição de uma s-rarefação e um s-choque.

A Figura 3.1, mostra a configuração das características no espaço  $x \times t$  quando  $t < t_o$ .

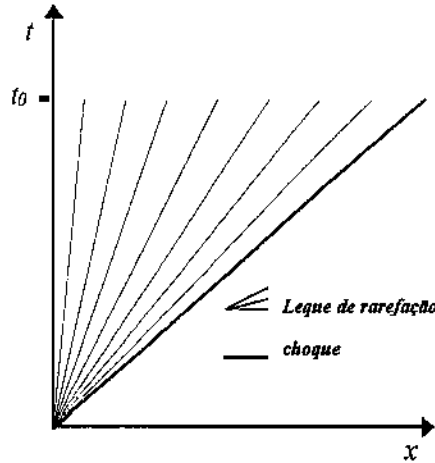


Figura 3.1 : Retas características

### 3.2.2. $t \geq t_0$ ( Interação entre s-onda e c-choque )

Quando  $t = t_0$  paramos de injetar água e começamos a injetar água com polímero. Temos portanto uma descontinuidade na concentração  $c(0, t)$  no ponto de injeção, pois passamos de  $c_{inj1} = 0$  para  $c_{inj1} > 0$ . Esta descontinuidade se propagará no interior do espaço  $x \times t$  como mostraremos logo mais. De acordo com Isaacson ([ISA]) a solução do problema de Riemann dependerá da localização dos estados à direita  $u_d = (s_d, c_d)$  e estados à esquerda  $u_e = (s_e, c_e)$  no plano das fases.

As seqüências de ondas compatíveis para o caso em que o valor da concentração à direita  $c_d$  seja menor que o da esquerda  $c_e$  e que o estado  $u_e$  pertence à região  $R$  é dada por:

$$u_e \xrightarrow{S} u^1 \xrightarrow{C} u_d \quad \text{se } s_d \geq s_k \quad (3.14)$$

$$u_e \xrightarrow{S} u^* \xrightarrow{C} u^2 \xrightarrow{S} u_d \quad \text{se } s_d < s_k \quad (3.15)$$

As Figuras 3.2 e 3.3 mostram a estrutura das ondas para as composições 3.14 e 3.15 respectivamente.

Como ilustrado na Figura 3.1 vamos considerar  $n$  retas características pertencentes ao leque de rarefação que compõe a solução da equação de Buckley-Leverett. De acordo com nossos testes o valor  $n = 200$  é satisfatório. Segundo o método das características cada reta carrega um valor  $s$  (saturação) e um valor  $c$  (concentração). A inclinação da característica é dada por  $\lambda^S(s, c) = f_s(s, c)$ . A reta que tem inclinação maior, no caso o eixo  $t$ , carrega o valor  $s = 1$  e a

característica de inclinação menor carrega o valor  $s = s_m$ . Vamos associar a estas retas um vetor  $s_d$ , com  $n$  coordenadas, sendo  $s_d(1) = 1$  e  $s_d(n) = s_m$  e considerar no plano das fases os estados  $u_d^j = (s_d(j), c_{inj1})$ ,  $j = 1, \dots, n$ . Para apresentarmos a solução é necessário que calculemos as curvas de discontinuidades decorrentes da injeção do segundo banco. Tais discontinuidades surgem a partir da interação entre ondas de rarefação e de choque. Passamos portanto descrever o algoritmo para a construção destas curvas, o que será feito em três etapas. Nas duas primeiras etapas descrevemos os procedimentos de cálculo dos dois primeiros pontos, a generalização do procedimento e o cálculo do último ponto. Na terceira etapa descrevemos as outras curvas de discontinuidades que surgem no problema.

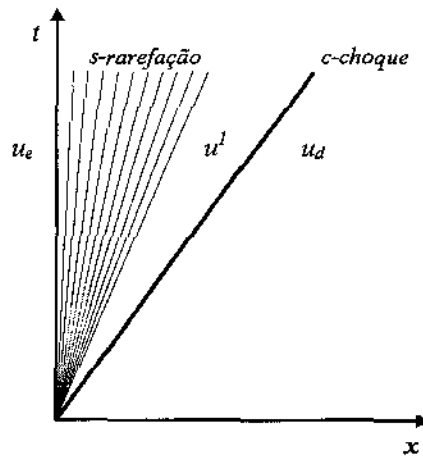


Figura 3.2 : Estrutura da onda para  $s_d \geq s_k$

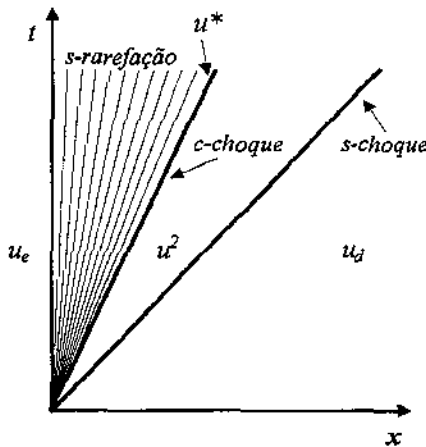


Figura 3.3 : Estrutura da onda para  $s_d < s_k$ .

**Etapa 1: Construção de  $y_{11}$ .**

**(1.A) : Cálculo do primeiro e segundo ponto** Denotemos por  $P_1$  o ponto inicial da curva de descontinuidade associada a mudança de bancos. Esta curva será denotada pelo par  $(y_{11}, t_{11})$ . Assim o ponto  $P_1$  é definido pelas coordenadas  $(y_{11}(1), t_{11}(1))$  que no nosso caso será  $(0, t_0)$ .

Consideremos o problema de Riemann associado a  $P_1$  com estados à direita e à esquerda dados respectivamente por:

$$u_d = u_d^2 = (s_d(2), 0) \text{ e}$$

$$u_e = u_{inj2} = (s_{inj2}, c_{inj2}) .$$

Como vimos a solução para o Problema de Riemann neste caso é do tipo (3.14). Na notação introduzida acima teremos:

$$u_{inj2} \xrightarrow{S} u_1^2 \xrightarrow{C} u_d^2 \tag{3.16}$$

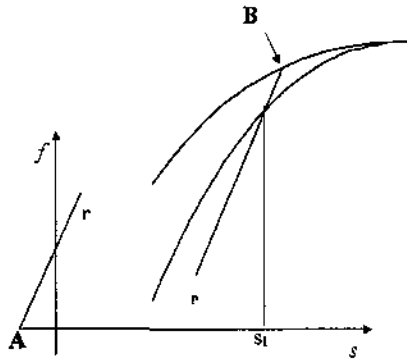


Figura 3.4 : Cálculo do estado intermediário  $u_1^2 = (s_1(2), c_{inj2})$ .

O estado intermediário  $u_1^2$  tem coordenadas  $(s_1(2), c_{inj2})$  onde o valor  $s_1(2)$  é a solução da equação:

$$\lambda^C(x, c_{inj2}) = \lambda^C(u_d^2) \tag{3.17}$$

sujeita a restrição  $\lambda^S(x, c_{inj2}) \leq \lambda^C(x, c_{inj2})$

Consideremos  $r$  a reta com inclinação dada por  $r$ , que passa por  $A = (-k, 0)$  e  $B = (s_d(2), f(s_d(2), 0))$ , onde  $k = \frac{s_{wi}}{s_n}$ . Geometricamente,  $s_1(2)$  é a ordenada do ponto de interseção da reta  $r$  com o gráfico da função de fluxo fracionário  $f(s, c_{inj2})$ , como mostra a Figura 3.4.

O ponto  $P_2$  é tomado como sendo a interseção da característica que carrega o estado  $u_d^2$ , com o c-choque dado em (3.15) que tem estado a esquerda  $u_1^2$ , como mostra a Figura 3.5.

Denotamos as coordenadas do ponto  $P_2$  pelo par  $(y_{11}(2), t_{11}(2))$ .

A solução final  $s(x, t)$  e  $c(x, t)$  para  $t_0 < t \leq t_{11}(2)$  é dada pela composição:

$$u_{inj2} \xrightarrow{S} u_1^2 \xrightarrow{C} u_d^2 \xrightarrow{S} u_o \quad (3.18)$$

**(1.B): Cálculo do terceiro ponto** Consideremos o problema de Riemann em  $P_2$ , com estado a esquerda  $u_e = u_1^2$  e estado a direita  $u_d = u_d^2$ . Se ainda tivermos que  $s_d(3) > s_k$  a solução do problema ainda é do tipo (3.14), ou seja, é dada pela composição:

$$u_1^2 \xrightarrow{S} u_1^3 \xrightarrow{C} u_d^3 \quad (3.19)$$

onde  $u_1^3$  é o estado intermediário calculado de maneira análoga à  $u_1^2$ .

O ponto  $P_3$  é tomado como a interseção da reta característica com inclinação com o c-choque que aparece na composição (3.19), como ilustra a Figura 3.6.

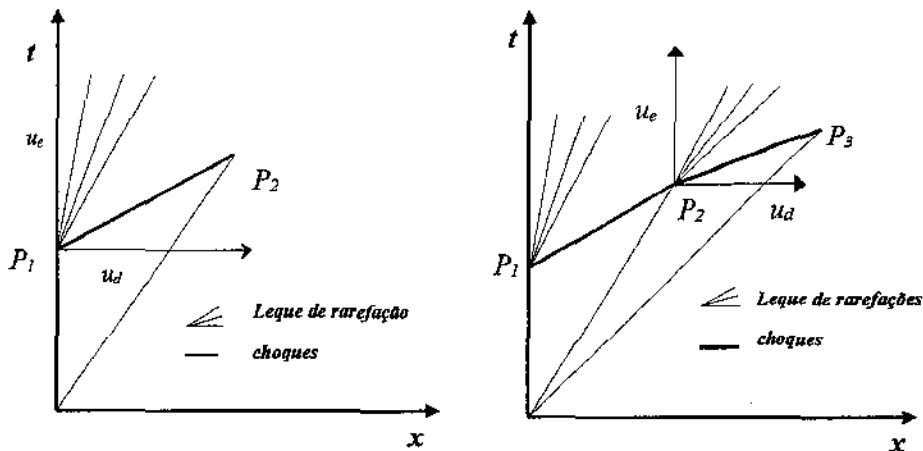


Figura 3.5 : Cálculo do ponto  $P_2$ . Figura 3.6 : Cálculo do ponto  $P_3$ .

A solução  $s(x, t)$  e  $c(x, t)$  para  $t_{11}(2) < t \leq t_{11}(3)$  é dada pela composição:

$$u_{inj2} \xrightarrow{S} u_1^3 \xrightarrow{C} u_d^3 \xrightarrow{S} u_o \quad (3.20)$$

**Observação:** Lembramos que a primeira s-onda que aparece é uma s-rarefação que liga os estados  $u_{inj2}$  ao estado  $u_1^3$ . A diferença entre (3.18) e (3.20) está no

valor da saturação de  $u_1^3$  ser menor que a de  $u_1^2$ . De maneira análoga, construímos os demais pontos da curva e paramos quando  $s_d(j) = s_k$  (podemos supor que existe um índice  $j_o \in \{1, \dots, n\}$  tal que  $s_d(j_o) = s_k$ ). De acordo com (3.14) este seria o caso limite, ou seja, o último ponto da curva  $y_{11}$ . A partir daí a composição da solução passa a ser do tipo (3.15). Sendo assim vamos calcular este ponto separadamente.

**(1.C): Cálculo do último ponto** Como  $s_d(j_o) = s_k$ , o problema de Riemann é tomado em  $P_{j_o-1}$  com estado a esquerda  $u_e = u_1^{j_o-1}$  e estado a direita  $u_d = (s_k, c_{in,j1})$ . A solução ainda é do tipo (3.14) e neste caso :

$$u_1^{j_o-1} \xrightarrow{S} u_1^{j_o} \xrightarrow{C} u_d^{j_o} \quad (3.21)$$

onde o estado intermediário  $u_1^{j_o} = u^*$  é calculado de maneira análoga ao detalhado quando calculamos  $u_1^2$ .

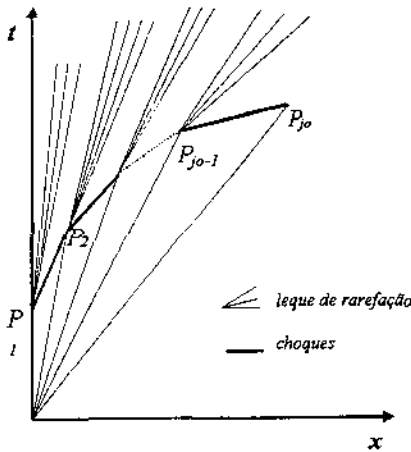


Figura 3.7

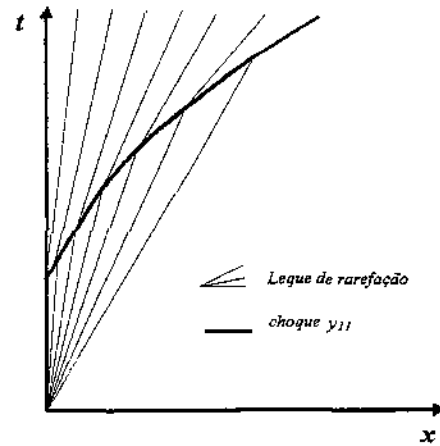


Figura 3.8

Calculamos então o ponto  $P_{j_o} = (y_{11}(j_o), t_{11}(j_o))$  de maneira análoga aos anteriores como mostra a Figura 3.7. A poligonal que liga os pontos  $P_1, P_2, \dots, P_{j_o}$ , aqui denotada por  $y_{11}$ , é portanto uma curva de choque com estados à direita dados por  $u_d^j$ ,  $j = 1, \dots, j_o$  e estados à esquerda  $u_e^1 (= u_1^j)$ , estes calculados de maneira análoga à  $u_1^2$  conforme Figura 3.4.

A solução para  $t_{11}(j_o - 1) < t \leq t_{11}(j_o)$  é dada pela composição:

$$u_{in,j2} \xrightarrow{S} u_1^{j_o} \xrightarrow{C} u_d^{j_o} \xrightarrow{S} u_o \quad (3.22)$$

Observamos aqui que a primeira s-onda é uma rarefação começando em  $u_{inj2}$  e terminando em  $u_1^{j_0}$  (esta rarefação sempre fará parte da composição da solução a partir de  $t = t_{11}(j_0)$ ).

A Figura 3.8 mostra a configuração parcial das características enquanto  $s_d(j) \geq s_k$ .

## Etapa 2: Construção da curva $y_{12}$

**(2.A) : Cálculo do primeiro e segundo ponto:** Tomemos  $Q_1$  com coordenadas  $(y_{12}(1), t_{12}(1))$  como sendo igual ao ultimo ponto da curva  $y_{11}$ . Portanto  $Q_1 = P_{j_0}$ . Consideremos então o problema de Riemann em  $Q_1$  com estado a esquerda  $u_e = u_1^{j_0}$  e estado a direita  $u_d = u_d^{j_0+1}$ . Como  $s_d(j_0 + 1) < s_k$ , a solução agora é do tipo (3.15), ou seja:

$$u^* \xrightarrow{C} u^2 \xrightarrow{S} u_d^{j_0+1} \quad (3.23)$$

Lembramos como foi descrito em (3.15) que a s-onda que liga  $u^2$  a  $u_d^{j_0+1}$  é um choque na saturação e que  $u^2 = (s_2, c_{inj1})$ , sendo  $s_2$  tal que,  $\lambda^C(s^2, c_{inj1}) = \lambda^S(s^*, c_{inj2})$ ,  $s_2 < s^*$ . Geometricamente, é dado pela abcissa do ponto de interseção da reta que passa por  $A = (s^*, f(s^*, c_{inj2}))$  e  $B = (-k, 0)$ ,  $k = \frac{s_{wi}}{s_n}$ , com o gráfico da função de fluxo fracionário  $f(s, c_{inj1})$  conforme Figura 3.9.

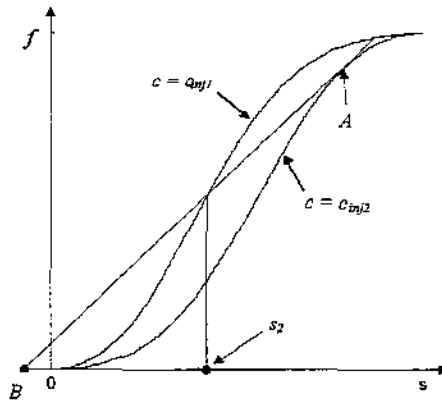


Figura 3.9 : Cálculo de  $u^2 = (s^2, c_{inj1})$ .

O segundo ponto da curva  $y_{12}$ ,  $Q_2 = (y_{12}(2), t_{12}(2))$  é obtido pela interseção da reta característica com inclinação  $\lambda^S(u_d^{j_0+1})$  com o s-choque de inclinação dada por  $m(s_d^{j_0+1})$  onde  $m(x)$  é dada por:

$$m(x) = \frac{f(x, c_{inj1}) - f(s^2, c_{inj1})}{x - s^2} \quad (3.24)$$

Na Figura 3.10 apresentamos a construção do ponto  $Q_2$  geometricamente.

A solução global  $s(x, t)$  e  $c(x, t)$  para  $t_{12}(1) < t < t_{12}(2)$  é dada pela composição:

$$u_{inj2} \xrightarrow{S} u^* \xrightarrow{C} u^2 \xrightarrow{S} u_d^{j_o+1} \xrightarrow{S} u_o \quad (3.25)$$

**(2.B): Cálculo do terceiro ponto:** Consideramos o problema de Riemann sobre  $Q_3$  com estado à esquerda  $u_e = u^2$  e estado a direita  $u_d = u_d^{j_o+2}$ . A solução é dada pelo s-choque:

$$u^2 \xrightarrow{S} u_d^{j_o+1} \quad (3.26)$$

Calculamos então o valor de  $Q_3 = (y_{12}(3), t_{12}(3))$  como sendo a interseção da reta característica com inclinação  $\lambda^S(u_d^{j_o+2})$  com o s-choque com inclinação  $m(u_d^{j_o+1})$ . A Figura 3.11 ilustra esta situação.

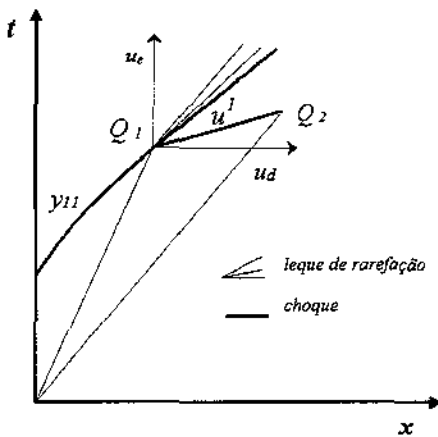


Figura 3.10 : O cálculo de  $Q_2$ .

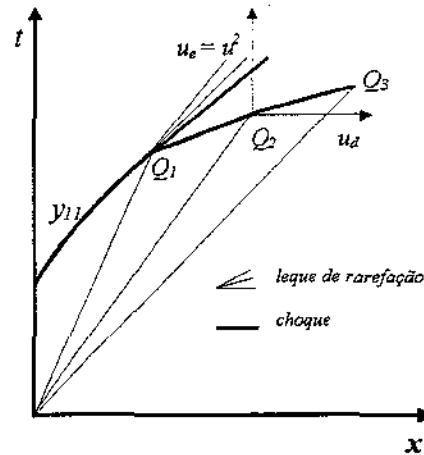


Figura 3.11 : O cálculo de  $Q_3$

**(2.C) : Cálculo do último ponto** Continuamos com o procedimento anterior até  $j = n$ . Neste caso teremos  $s_d^n = s_m$ . Consideramos o problema de Riemann em  $Q_{n-1} = (y_{12}(n-1), t_{12}(n-1))$  com estados à esquerda e à direita dados por  $u_e = (s_2, c_{inj1})$  e  $u_d = (s_d^n, c_{inj1})$ .



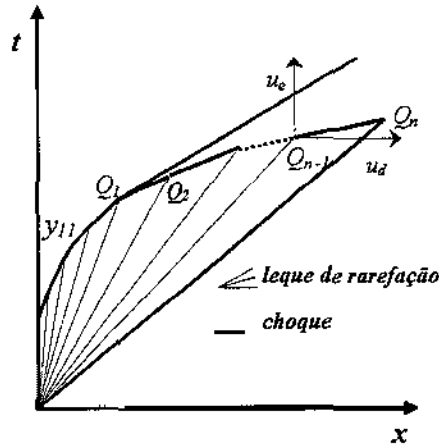


Figura 3.12 : Cálculo de  $Q_n$

O ponto  $Q_n$  é obtido pela interseção da reta característica com inclinação com inclinação  $\lambda^s(u_d^n)$  com a reta(curva de choque ) com inclinação dada por  $m(u_d^n)$  como podemos ver na figura 3.12.

Observamos que a poligonal que une os pontos  $Q_1, Q_2, \dots, Q_n$  é um s-choque e de acordo com a solução dada em (3.15) o estado à esquerda desta curva em qualquer ponto será  $u^2 = (s^2, c_{in,j1})$ . Pela própria construção ela será denotada por  $y_{12}$ .

### Etapa 3: Construção da curva $y_{13}$ , $y_{14}$ e $y_o$

a) **curva  $y_{13}$ :** Consideremos o problema de Riemann em  $Q_n$  com estado a esquerda  $u_e = u_2$  e estado a direita  $u_d = u_o$ . A solução é dada por um s-choque, neste caso:

$$u^2 \xrightarrow{s} u_o \quad (3.27)$$

Este choque é uma reta que se propaga a partir de  $Q_n$  e será denotada por  $y_{13}$  ( ver Figura 3.13).

b) **curva  $y_{14}$**  A solução está bem definida acima das curvas  $y_{11}$  e  $y_{12}$  pelas características do leque de rarefação que se formam a partir delas. Dentre elas um outro choque aparece naturalmente na solução final, já que a partir de  $t = t_{11}(j_o)$  o c-choque:

$$u^* \xrightarrow{c} u^2 \quad (3.28)$$



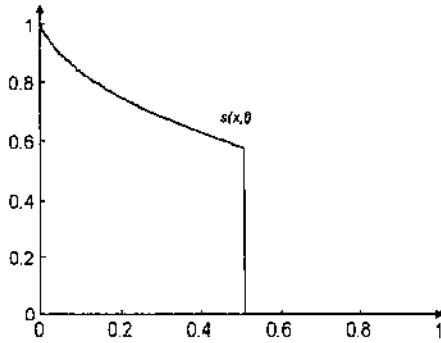


Figura 3.14

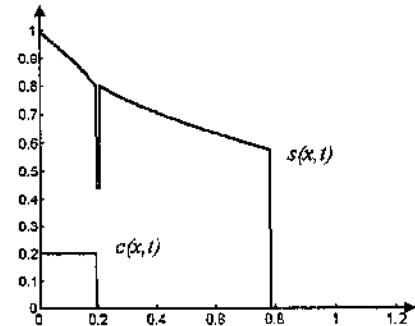


Figura 3.15

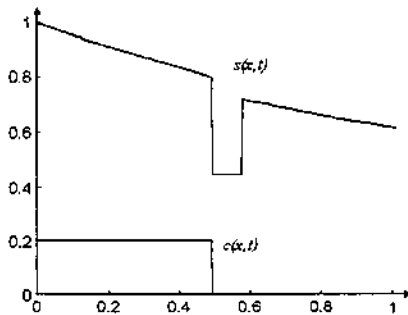


Figura 3.16

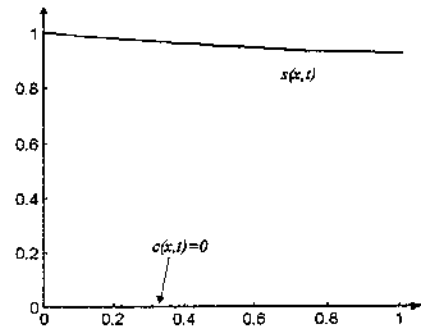


Figura 3.17

### 3.3. A injeção de três bancos

O modelo matemático da injeção de um terceiro banco, consiste em alterarmos a condição de contorno, que agora fica da seguinte forma:

$$\begin{cases} s(0,t) = 1 & c(0,t) = c_{inj1} & 0 \leq t < t_0 \\ s(0,t) = 1 & c(0,t) = c_{inj2} & t_0 \leq t < t_1 \\ s(0,t) = 1 & c(0,t) = c_{inj3}, & t_1 \leq t \leq 1 \end{cases} \quad (3.29)$$

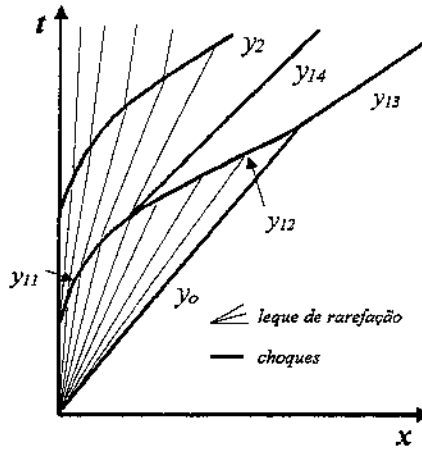


Figura 3.18 : Configuração final das características para o caso de três bancos.

Supondo que a partir de  $t = t_1$ , paramos de injetar água com polímero e voltamos a injetar água sem polímero, devemos ter  $c_{inj1} = 0$ ,  $c_{inj2} > 0$  e  $c_{inj3} = 0$ . Olhando a configuração final das características para o caso em que dois bancos são injetados (ver Figura 3.13), observamos que se injetamos um terceiro banco à partir de  $t = t_1$  uma nova curva de choque se forma pela interação entre as características já existentes na configuração associado a  $t < t_1$  e as características associadas a injeção do novo banco. Esta nova curva será denotada por  $y_{21}$  e sua construção é feita de maneira análoga a curva  $y_{11}$ , ou seja, considerando problemas de Riemann locais. A Figura 3.18 mostra uma configuração para as rarefações e choques no caso da injeção de três bancos.

Se injetarmos novos bancos, novas curvas de descontinuidades aparecerão e o tratamento será análogo. Sendo assim a solução semi-analítica para o caso sem adsorção fica bem definida. Esta solução depende exclusivamente do tratamento dado as descontinuidades do problema.

### 3.4. Modelo com Adsorção - Injeção de dois bancos.

O caso da injeção de dois bancos considerando o fator adsorção consiste em resolver o sistema (3.2) com condições iniciais e de fronteira do tipo (3.13)

Supondo que o primeiro banco injetado seja de água e o segundo seja de água com polímero, teremos  $c_{inj1} = 0$  e  $c_{inj2} > 0$ . As descontinuidades são calculadas de maneira análoga ao caso sem adsorção, pois as c-ondas quando  $c_{inj1} < c_{inj2}$  serão sempre c-choques e as s-ondas soluções da equação de Buckley-Leverett.

O caso em que a concentração do segundo banco é menor que a concentração do primeiro banco torna-se complicado quando tentamos aplicar a mesma técnica utilizada nos casos anteriores. Tal complicação surge da dificuldade de analisarmos a interação entre uma s-rarefação e uma c-rarefação que aparece na solução do problema de Riemann apresentado no capítulo 2. Este é um problema interessante para ser estudado futuramente.

Com base nestes algoritmos elaboramos um programa que calcula estas curvas de choque e as saturações e concentrações para cada tempo  $t$ . Além disso o programa calcula a produção de óleo durante qualquer período de injeção de fluidos no reservatório.

Nosso objetivo no próximo capítulo será utilizar este programa para fazermos comparações entre os processos de recuperação citados até aqui e também analisar as vantagens do método semi-analítico descrito (na simulação do processo) em relação a métodos numéricos padrões para problemas deste tipo.

Lembramos também que se tomarmos a saturação de injeção no segundo banco  $s_{inj2} < 1$  outra descontinuidade aparece e pode ser calculada utilizando a mesma técnica utilizada no cálculo de  $y_{11}$ . O programa desenvolvido não leva em conta esta possibilidade. Ele foi desenvolvido tomando saturações de injeção dos bancos sempre iguais à 1.

## 4. TESTES NUMÉRICOS

Neste capítulo vamos apresentar os resultados obtidos da análise dos experimentos numéricos fornecidos pelo programa computacional desenvolvido. Este programa simula a solução para o problema da recuperação de petróleo usando aditivos ativos à água injetada no reservatório.

Nos nossos programas utilizaremos a seguinte expressão algébrica para simular a função de fluxo fracionário:

$$f(s, c) = \frac{s^2}{s^2 + (0.5 + 100.c).(1 - s)^2} \quad (4.1)$$

para o caso da injeção de polímeros ( polymer-flooding). No caso da injeção de água sem polímeros, tomamos  $c = 0$  nesta expressão e temos:

$$f(s) = \frac{s^2}{s^2 + 0.5(1 - s)^2} \quad (4.1)$$

De acordo com Lake ( [LAK} ) o fenômeno da adsorção será simulado pela expressão:

$$a(c) = \frac{k_1.c}{1 + k_2.c} \quad (4.2)$$

que dependerá dos parâmetros  $k_1$  e  $k_2$  que também serão fixados. No nosso caso tomaremos  $k_1 = 0.2$  e  $k_2 = 100$ .

### 4.1. Simulações no caso sem adsorção.

Na simulação do processo de recuperação de óleo o que nos interessa é calcular o valor da saturação de óleo em  $x = 1$ , que de acordo com nossa adimensionalização representa a localização do poço produtor, ou seja, onde mediremos a quantidade de óleo que está sendo produzida pelo processo.

Como nossos programas foram elaborados tendo como incógnita a saturação de água,  $s(x, t)$ , o cálculo da quantidade de óleo produzida como função do tempo é medida, de acordo com Marle ([MAR]) pela integral:

$$Q_o(t) = \int_0^t (1 - f(s(1, \tau), c(1, \tau))) d\tau \quad (4.3)$$

que representa o acumulado de óleo recuperado expresso como uma fração do volume poroso.

Para os testes que apresentaremos nos experimentos a seguir, vamos considerar um reservatório com as seguintes características: saturação de água inicial  $s_{wi} = 0.1$  (correspondendo a 10% do volume total); saturação de óleo residual  $s_{or} = 0.1$  e concentração de polímero  $c = 0$

### Experimento 1: Injeção de água com polímero - sem adsorção.

Vamos injetar a mistura água com polímeros no reservatório com saturação de injeção  $s = 1$  e com concentração de polímeros  $c = 0.01$ .

O problema de Riemann neste caso é dado pelo sistema de equações (3.1) com estados à esquerda e à direita dados por:

$$\begin{cases} s(x, 0) = s_d = 0, & c(x, 0) = c_d = 0 \quad (\text{estado à direita}) \\ s(0, t) = s_e = 1, & c(0, t) = c_e = 0.01 \quad (\text{estado à esquerda}) \end{cases}$$

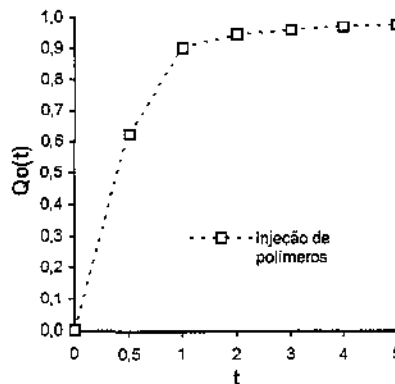


Figura 4.1 : Evolução da produção de óleo  $Q_o(t)$  para o processo de injeção de água com polímeros.

O programa desenvolvido calcula o valor da saturação  $s(x, t)$  baseado na solução analítica do problema de Riemann, apresentada por Isaacson (ver [ISA])) e consiste da composição c-ondas e s-ondas ( capítulo 2 ). A Figura 4.1 mostra que a quantidade de óleo recuperado quando um volume poroso é injetado,  $t_D = 1$ , é da ordem de 90 %. Para  $t_D = 5$  a recuperação é da ordem de 97%.

Igualmente ao caso anterior temos uma convergência assintótica do gráfico da produção para o valor máximo, que no caso será  $Q_o = 1$ . Observamos porém que esta convergência é mais rápida quando comparada com a injeção de água (water-flooding ). A vantagem da injeção de polímero está no fato de que recuperamos uma quantidade de óleo maior num espaço de tempo menor. Esta conclusão está ilustrada no gráfico da Figura 4.2 na qual apresetamos os resultados da recuperação nos casos da injeção de água (water-flooding) ou água com polímeros (polymer-flooding).

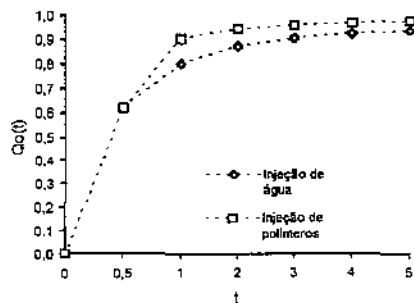


Figura 4.2 : Comparação dos gráficos de evolução da produção para a injeção de água e a injeção de polímeros.

### Experimento 2: Influência da concentração de polímero na produção

Aqui vamos variar a concentração de polímero  $c$  misturado à água no caso sem adsorção. De acordo com a literatura ( ver [LAK] ) um valor máximo para a concentração está próximo de  $c = 0.05$ . A título de ilustração calculamos até o valor  $c = 0.1$ .



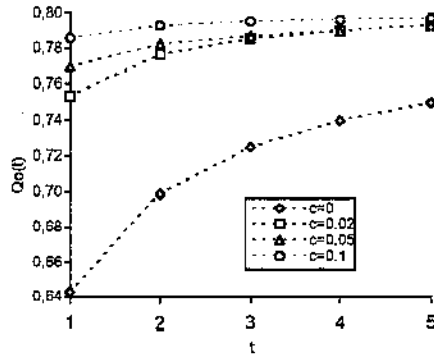


Figura 4.3 : Influência da concentração de polímero  $c$  na água durante o processo de injeção.

Nesta simulação observamos um aumento da ordem de 18% na produção quando um volume poroso de fluido é injetado, se passamos da concentração  $c = 0$  para a concentração  $c = 0.1$ , ou seja neste mesmo período de tempo ( $t_D = 1$ ) estamos recuperando uma quantidade de óleo 18% maior. Novamente observamos que o gráfico tende a convergir para o valor máximo (100% que corresponde à  $Q_o = 1$ ), quando  $t$  cresce. Uma quantidade maior de polímero acarretará portanto em uma recuperação mais rápida. Isto está evidenciado na figura 4.3 onde apresentamos as recuperações obtidas nos tempos  $1 \leq t \leq 5$ , em volumes porosos. A título de comparação usamos, nesta figura,  $c = 0$ ,  $c = 0.02$ ,  $c = 0.05$  e  $c = 0.1$ .

Lembramos que estes resultados foram obtidos utilizando-se a função de fluxo fracionário dada pela expressão (4.1) no início do capítulo. Entretanto, como observado por Isaacson ([ISA]) a eficiência da recuperação depende das propriedades da função de fluxo fracionário e das características iniciais do reservatório.

## 4.2. Simulações no caso com adsorção.

Neste caso, no sistema de equações diferenciais parciais que modela o deslocamento do óleo, aparece a função  $a(c)$  dada em (4.3) que modela a adsorção. O sistema neste caso é dado por:

$$\begin{cases} s_t + f(s, c)_x = 0 \\ (s \cdot c + a(c))_t + (c \cdot f(s, c))_x = 0 \end{cases}$$

### Experimento 3 : Injeção de água com polímeros - com adsorção

Com os dados dos experimentos anteriores calculamos a produção de óleo quando injetamos água com polímeros, utilizando o modelo que leva em conta o efeito da adsorção.

O gráfico da Figura 4.4 mostra a evolução da produção de óleo durante o período de injeção de fluidos, quando injetamos água e quando injetamos água com polímeros no caso com adsorção.

Nota-se que a partir de  $t_D = 2$  a produção tende ao valor máximo em  $Q_o = 1$  (corresponde a 100% do óleo recuperável) e que a recuperação é mais rápida para o caso da injeção de água com polímeros.

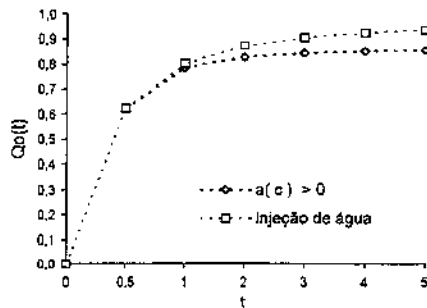


Figura 4.4 : Evolução da produção de óleo  $Q_o(t)$  para o caso da injeção de polímeros com adsorção.

### 4.3. Simulações no caso da injeção de bancos.

Com o auxílio do algoritmo apresentado no capítulo anterior elaboramos um programa computacional que calcula a solução para o problema da injeção de bancos. Através deste programa vamos fazer alguns experimentos que nos mostrem as vantagens deste processo sobre os processos de injeção de um banco de água (waterflooding) ou da injeção de água com polímeros ( polimer-flooding ).

Para nossas simulações iniciais vamos ainda considerar que o reservatório tem saturação de água inicial  $s_{wi} = 0.1$  e saturação de óleo residual  $s_{or} = 0.1$  e concentração de polímero  $c = 0$ .

#### Experimento 4: Injeção de 2 bancos - Caso sem adsorção

Vamos considerar a situação em que injetamos 20% de um volume poroso de água no reservatório e a partir daí injetamos a mistura de água com polímero com uma concentração  $c = 0.01$ . De acordo com os nossos cálculos, injetar 20% do volume poroso de água corresponde à tomarmos  $t_o = \frac{0.2}{s_n} (s_n = 1 - s_{wi} - s_{or})$  na solução do sistema de equações parciais adimensional.

Assim as seguintes condições iniciais e de contorno para o sistema são dadas por:

$$\begin{cases} s(x, 0) = 0 & c(x, 0) = 0 & 0 < x \leq 1 \\ s(0, t) = 1 & c(0, t) = 0 & 0 < t \leq t_o \\ s(x, 0) = 1 & c(x, 0) = 0.01 & t_o < t \end{cases}$$

Na Figura 4.5 mostramos os perfis da saturação calculada no programa computacional para os valores de  $t = t_2$  e  $t = t_3$ , sendo que  $t_2 < t_3$ .

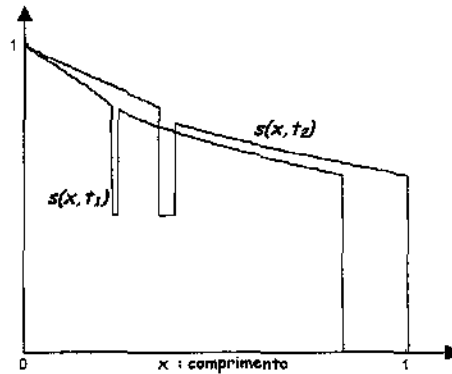


Figura 4.5 : Perfil da saturação  $s(x, t)$  para dois valores de  $t$ .

Utilizando o programa computacional calculamos a evolução da produção de óleo até que 5 volumes porosos de fluido sejam injetados, ou seja até  $t_D = 5$ . Os cálculos mostram (ver Figura 4.6) que a produção converge para a quantidade de óleo máxima  $Q_o = 1$ .

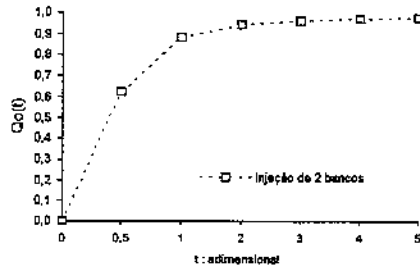


Figura 4.6 : Evolução da produção de óleo  $Q_o(t)$  durante o período de injeção dos bancos de água e água com polímeros.

Na Figura 4.7 comparamos a evolução da produção, quando injetamos somente água com polímeros e quando injetamos um banco de água seguido de água com polímeros. Observamos que a produção é um pouco maior no início ( $t_D = 1$ ), mas no final é praticamente a mesma. Matematicamente poderíamos dizer que temos uma vantagem com relação a custo de produção, já que gastamos menos polímeros para recuperar a mesma quantidade de óleo em um mesmo espaço de tempo.

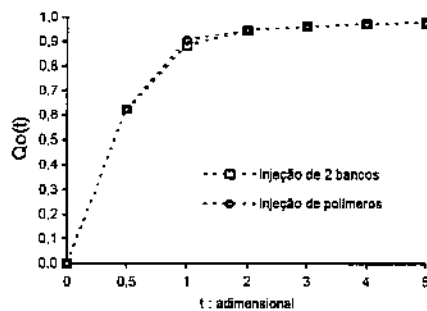


Figura 4.7 : Evolução da produção de óleo por meio da injeção de polímeros e a injeção de bancos.

### Experimento 5: Influência do valor de $t_0$ na produção

No caso da injeção de bancos, o valor de  $t_0$  (tamanho do primeiro banco) desempenha papel importante para uma análise da produção de óleo. Vamos destacar este fato neste experimento, analisando a produção de óleo depois que um volume poroso de fluido foi injetado ( $t_D = 1$ ) para o caso sem adsorção. Para o caso com adsorção os resultados serão análogos.

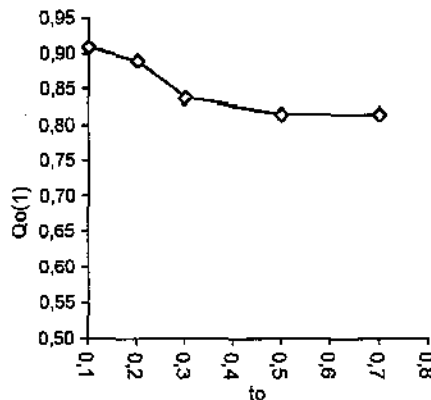


Figura 4.8 : A produção  $Q_o(1)$  em função da variação do valor de  $t_o$ .

Fazendo variar o valor de  $t_o$ , verificamos que a quantidade de óleo produzida é para  $t_o = 0.1$  é maior que a quantidade produzida para  $t_o = 0.8$ . O fato é que quando  $t_o = 0.1$  estamos injetando pouca água e bastante polímero e a produção é dominada pela injeção de polímero. Para  $t_o = 0.8$ , o processo está sendo dominado pela injeção de água que recupera uma quantidade menor. A Figura 4.8 mostra o gráfico da produção de óleo  $Q_o(1)$  (um volume poroso injetado) para diversos valores de  $t_o$ .

Observamos através do gráfico, que durante o processo de recuperação, se já tivermos injetado 50% do volume poroso de água no reservatório não seria interessante injetarmos a mistura de água com polímero, pois isso não acarretará em um aumento da produção de óleo quando  $t_D = 1$ .

### Experimento 6: A injeção de 2 bancos para o caso com adsorção.

Considerando os dados do Experimento 4 calculamos a produção de óleo quando injetamos dois bancos, sendo o primeiro de água e o segundo de água com polímeros, no caso em que o fator adsorção é considerado.

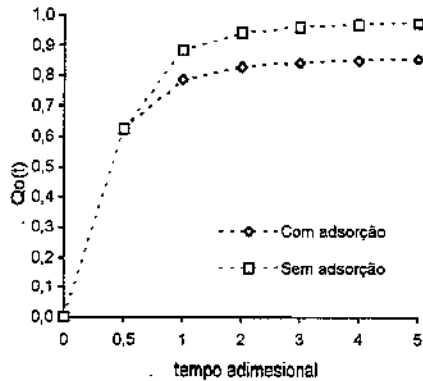


Figura 4.9 : A evolução da produção de óleo  $Q_o(t)$  quando injetamos 2 bancos para os modelos com adsorção e sem adsorção.

No gráfico da Figura 4.9 apresentamos a comparação da evolução produção nos casos da injeção de dois bancos para os modelos com e sem adsorção. Nele podemos concluir que a produção para o caso com adsorção é mais lenta quando comparada com o caso sem adsorção.

### Experimento 7: A injeção de 3 bancos para o caso sem adsorção

Vamos considerar agora o processo de recuperação que consiste da injeção de três bancos, sendo o primeiro de água, o segundo de água com polímero e o terceiro novamente de água. Vamos considerar o modelo sem adsorção proposto por Isaacson. Como ilustração consideremos a injeção de fluidos no reservatório distribuídos da seguinte maneira: 20% de água no primeiro banco, seguida de mais 10% da mistura de água e polímero, este com concentração  $c = 0.01$  e depois volta-se a injetar água. Para efeito de cálculo estamos supondo que o reservatório tenha saturação de água inicial  $s_{wi} = 0.1$  e saturação de óleo residual  $s_{or} = 0.1$ . Esta situação corresponde a tomarmos as seguintes condições iniciais e de contorno para o modelo matemático:

$$\begin{cases} s(x, 0) = 0 & c(x, 0) = 0 & 0 < x \leq 1 \\ s(0, t) = 1 & c(0, t) = 0 & 0 < t \leq t_o \\ s(0, t) = 1 & c(0, t) = 0.01 & t_o < t \leq t_1 \\ s(0, t) = 1 & c(0, t) = 0 & t_1 < t \end{cases}$$

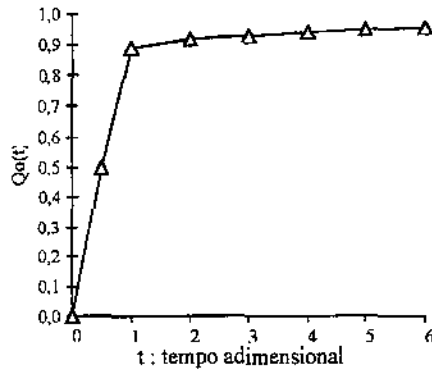


Figura 4.10 : Evolução da produção quando injetamos 3 bancos.

Notamos através do gráfico da Figura 4.10 que a produção associada à injeção de um volume poroso no reservatório é de aproximadamente 89% da quantidade total. Quando  $t_D = 5$  o gráfico mostra que a produção tende a se estabilizar e a recuperação é mais lenta.

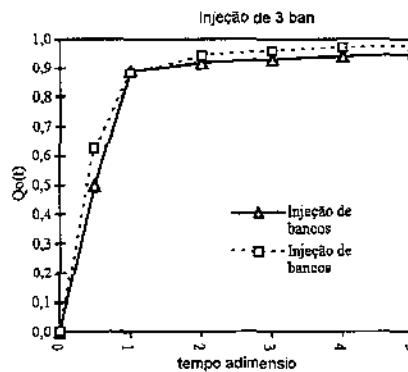


Figura 4.11 : Comparação da evolução da produção de óleo  $Q_o(t)$  quando injetamos 2 e 3 bancos.

O programa desenvolvido calcula o valor da saturação  $s(x,t)$  baseado na solução analítica do problema de Riemann, apresentada por Isaacson (ver [ISA]) e consiste da composição c-ondas e s-ondas ( capítulo 2 ). A Figura 4.1 mostra que

a quantidade de óleo recuperado quando um volume poroso é injetado,  $t_D = 1$ , é da ordem de 90 %. Para  $t_D = 5$  a recuperação é da ordem de 97%.

Lembramos mais uma vez, que como usamos a teoria do fluxo fracionário o comportamento da função fluxo fracionário é crucial.

#### 4.4. O método numérico e o semi-analítico

Consideremos o seguinte esquema numérico (ver [TVT]) para a solução do sistema de equações diferenciais parciais associados aos modelos sem adsorção :

$$\begin{aligned} s_j^{n+1} &= s_j^n - \alpha(f_j^n - f_j^n) \\ c_j^{n+1} &= c_j^n - \alpha g_j^n (c_j^n - c_j^n) \end{aligned} \quad (4.4)$$

Nesta notação,

$$(s_j^n, c_j^n) = (s(j.\Delta x, n.\Delta t), c(j.\Delta x, n.\Delta t)) \quad \forall (j, n) \in Z^+ \times Z^+$$

denota a solução aproximada do modelo em questão e

$$\begin{aligned} \alpha &= \frac{\Delta x}{\Delta t} \\ f_j^n &= f(s_j^n, c_j^n) \\ g_j^n &= g(s_j^n, c_j^n) \end{aligned}$$

A função  $g(s, c)$  é dada por:  $g(s, c) = \frac{f(s, c)}{s}$  para o modelo sem adsorção e  $g(s, c) = \frac{f(s, c)}{s + h(c)}$  para o modelo com adsorção.

Assumimos que os parâmetros da malha satisfazem a condição CFL (Condição de Courant-Friedrichs-Lewy) dada por  $CFL = \frac{\Delta x}{\Delta t} \sup_{(s,c) \in I \times I} \left\{ \frac{\partial f}{\partial s}, g \right\} \leq 1$ . Para a função de fluxo fracionário que estamos utilizando ela é dada pelo valor  $CFL = \frac{12}{25}$ .

Utilizando o esquema numérico dado acima e o programa computacional que implementa a solução analítica do problema, vamos comparar as soluções nos casos citados anteriormente.

##### A) Injeção de água com polímeros sem adsorção

Consideremos o caso da injeção de água com polímeros sem adsorção.



Na Figura 4.12 apresentamos os perfis da solução calculada utilizando o método numérico citado e o método semi-analítico descrito nos capítulos anteriores. Notamos que a solução numérica descreve bem a parte contínua da solução analítica. Nas descontinuidades porém observamos que a solução numérica apresenta dispersão, suavizando a solução naquele ponto. Na figura o valor da concentração está multiplicado por 10 para que possamos ter uma melhor visualização.

### B) Injeção de 2 bancos - Caso sem adsorção

Neste caso o problema apresenta três descontinuidades. Verificamos que a solução numérica também suaviza as descontinuidades. A Figura 4.13 mostra os perfis de saturação e concentração das soluções calculadas através dos esquemas numérico e semi-analítico.

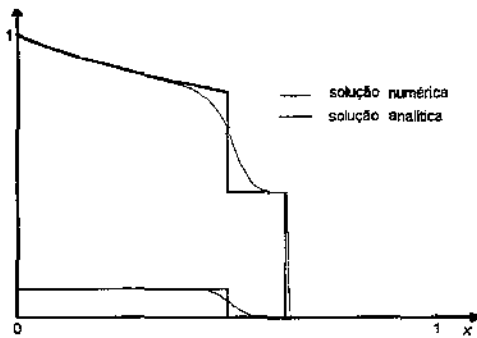


Figura 4.12

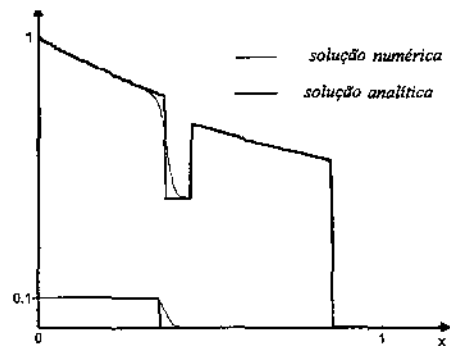


Figura 4.13

### C) Injeção de 3 bancos - Caso sem adsorção

Neste caso a solução apresenta quatro descontinuidades na solução mostradas na Figura 4.14. Como nos casos anteriores a solução numérica suaviza estas descontinuidades deixando de mostrar certas particularidades da solução.

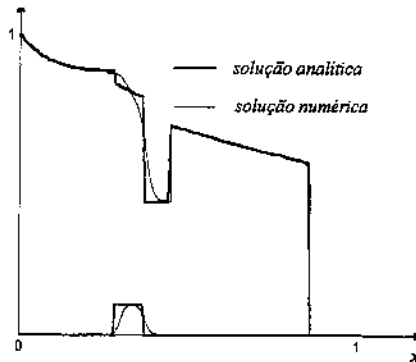


Figura 4.14

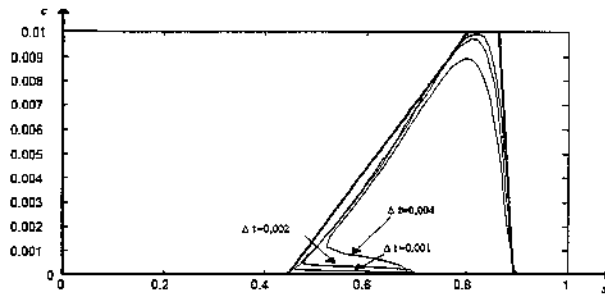


Figura 4.15 : Convergência da solução numérica para a solução semi-analítica no plano das  $s \times c$ .

Na Figura 4.15 apresentamos os caminhos da solução obtida no espaço de estados. Podemos notar que as soluções numéricas em  $s \times c$ , converge para a solução semi-analítica apresentada. De acordo com Le Veque ([LEV]) e Crandall ( ver [CRA]) podemos afirmar que a solução semi-analítica obtida é a solução fisicamente (ou solução de entropia) correta uma vez que o método numérico utilizado é um método de primeira ordem monótono e convergente. Este teste serve como validação da solução que apresentamos no capítulo 3.

### E) Métodos semi-analítico e numérico quanto ao tempo computacional

O fato importante a ser destacado é o tempo computacional gasto para a simulação do processo. Observamos que apesar de o método semi-analítico ter uma implementação computacional mais elaborada, o tempo computacional gasto para apresentarmos a solução é muito menor que o gasto pelo esquema numérico proposto, além disso a solução semi-analítica descreve as descontinuidades da solução com mais precisão (já que as descontinuidades são suavizadas no método numérico).

A Tabela abaixo compara os dados obtidos para a solução do Problema de Riemann associado a injeção de polímeros, caso sem adsorção. Ela mostra que o tempo gasto pelo método semi-analítico para obtenção da solução é bem inferior ao do método numérico. Por exemplo se tomamos uma malha com 100 pontos, observamos que o método numérico gasta mais que o dobro do tempo para a obtenção da solução. Para o caso de 1000 pontos, o método numérico chega a gastar mais de 150 vezes o tempo gasto pelo método semi-analítico. Tais medidas de tempo foram realizadas utilizando recursos do programa MATLAB.

$n^{\circ}$ de pontos	tempo gasto pelo Método semi-analítico	tempo gasto pelo Método Numérico	
100	3	8	
200	4	34	
300	4	77	
400	4	136	
500	4	214	(4.5)
600	4	312	
700	5	416	
800	5	546	
900	5	693	
1000	5	854	

Podemos dizer que o método semi-analítico é mais vantajoso em vários aspectos como o tempo gasto para obtenção da solução (conforme tabela) sendo portanto recomendável para problemas que exijam repetidos cálculos da solução como por exemplo no cálculo da injeção de bancos 2D configuração 5-spot no reservatório usando a teoria dos canais de fluxo. Outro aspecto importante é uma melhor reprodução das frentes de descontinuidades que aparecem no problema.

## 5. CONCLUSÕES

O propósito deste trabalho foi a implementação e elaboração de algoritmos para a solução numérica do sistema de equações que modela o problema da recuperação de petróleo através da injeção de água, injeção de água com polímeros e principalmente o da injeção de bancos.

Para o problema da injeção de bancos, que é caracterizado por um sistema de equações diferenciais parciais com condições iniciais e de contorno dadas, desenvolvemos um método, onde utilizamos as soluções do problema de Riemann para a localização das curvas de descontinuidades que aparecem no problema. Esta também é a idéia dos métodos Front-Tracking. A diferença é que nas regiões de suavidade utilizamos também o método das características.

Para a elaboração a apresentação dos algoritmos utilizamos as soluções apresentadas por Isaacson [ISA], Johansen [JOH] e também as idéias lançadas por Barenblatt [BRB] e Bedrikovetsky [BED]. Os algoritmos localizam as descontinuidades da solução no plano  $x \times t$ , como ilustrado na Figura 5.1 através do estudo da interação entre ondas de choque e de rarefação que aparecem na solução do problema de Riemann.

Com a implementação deste algoritmo, fizemos simulações de situações práticas associadas à injeção de bancos. Ilustramos através de exemplos que é possível analisar as vantagens dos processos de recuperação quando comparamos a produção de óleo. Alguns resultados já conhecidos são confirmados, como a melhora da eficiência da recuperação quando utilizamos o processos da injeção de bancos. Observamos porém que vários fatores influenciam esta melhora, como por exemplo o tamanho dos bancos (valor de  $t_o$ ), as condições iniciais do reservatório ( $s_{wi}$  e  $s_{or}$ ) e a quantidade de polímero (concentração  $c$ ) adicionada a água. A função fluxo fracionário também exerce influência no processo de recuperação, uma vez que para fluidos com características diferentes os resultados podem ser outros.

Comparamos o método semi-analítico estudado com um método numérico de primeira ordem ([TVT]). Nesta comparação analisamos o comportamento da solução no plano das fases  $s \times c$  e vimos através de exemplos que a solução obtida

pelo método numérico converge para a solução semi-analítica. De acordo com Crandall [CRA] e Le Veque [LEV] esta convergência garante que a solução dada pelo algoritmo é a solução de entropia ou solução fisicamente correta.

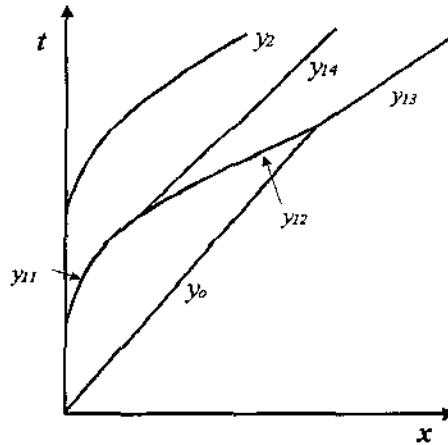


Figura 5.1 : As descontinuidades da solução para o problema da injeção de dois bancos

A implementação da solução utilizando o método semi-analítico é elaborada, pois envolve o estudo da solução do Problema de Riemann associado. Usamos a comparação com o método de primeira ordem citado para ilustrar que a solução semi-analítica é obtida com um tempo computacional bem inferior. Este fato está explicitado no capítulo 4, seção 4.4 resumido na tabela (4.5). Entretanto neste trabalho não fizemos comparações com métodos numéricos mais eficientes como por exemplo os métodos de alta resolução.

Os programas computacionais foram elaborados dentro do ambiente MATLAB, o qual possui uma linguagem própria próxima da linguagem C. Várias rotinas do MATLAB facilitaram a implementação do nosso algoritmo. Destaque também para a parte gráfica que nos permitiu uma melhor visualização das soluções calculadas.

## 5.1. Trabalhos Futuros

Pretendemos continuar o estudo do caso com adsorção. De fato, neste trabalho não conseguimos aplicar as técnicas apresentadas no capítulo 3, para estabelecer a solução do problema da injeção de bancos quando a concentração de polímero

injetada no segundo banco é menor que a concentração injetada no primeiro banco. Neste caso a dificuldade se deve à solução do problema de Riemann que apresenta c-rarefações o que não acontece no caso sem adsorção.

Podemos ainda aplicar a técnica apresentada no capítulo 3 para outros processos relacionados com a produção de petróleo.

Numa próxima fase de nossos estudos pretendemos aplicar métodos numéricos de alta resolução no problema da injeção de bancos e compará-los com o método semi-analítico estudado.

Vamos aprimorar o programa computacional desenvolvido de modo que possamos varrer todas as possibilidades, uma vez que neste trabalho elaboramos os programas para alguns casos particulares.

Outra possibilidade é abordar o caso bi-dimensional, de maior interesse do ponto de vista prático. Cabe destacar aqui que a solução semi-analítica apresentada neste trabalho poderá ser usada neste caso utilizando a teoria dos chamados canais de fluxo.

## 6. APÊNDICE A

### 6.1. A descrição do modelo

A obtenção do modelo matemático para a injeção de água com aditivos é baseada sobre considerações de balanço de material ([ISA],[BRB]). Assumindo que fluido e rocha são incompressíveis, que os volumes não mudam quando o polímero é dissolvido na água, o princípio de conservação de massa de água, óleo e aditivo num escoamento bifásico unidimensional nos conduz às seguintes equações:

$$\phi.s_t + \nu_x = 0 \quad (6.1)$$

$$\phi.s_t^o + \nu_x^o = 0 \quad (6.2)$$

$$\phi.(s.c)_t + (\nu.c)_x = 0 \quad (6.3)$$

Nestas equações,  $s$  (respectivamente  $s^o$ ) representa a saturação de água (saturação do óleo),  $\nu$  (respectivamente  $\nu^o$ ) a taxa de escoamento volumétrico da fase água ( óleo ),  $\phi$  é a porosidade da rocha ( toamda constante ) e  $c$  denota a concentração de aditivo na água. Ignorando os efeitos de gravidade, capilaridade e dispersão, temos pela Lei de Darcy:

$$\nu = -\lambda.p_x \quad (6.4)$$

$$\nu^o = -\lambda^o.p_x \quad (6.5)$$

onde

$$\lambda = \frac{K.k}{\mu} \quad (6.6)$$

$$\lambda^o = \frac{K.k^o}{\mu^o} \quad (6.7)$$

Nas equações acima  $p$  é a pressão do fluido,  $K$  é a permeabilidade absoluta da rocha,  $k = k(s, c)$  e  $k^o = k^o(s)$  são as permeabilidades relativas da fase água e óleo respectivamente,  $\mu, \mu^o$  são as correspondentes viscosidades.

Somando (6.1) com (6.2) temos:

$$\phi.(s_t + s_t^o) + (\nu_x + \nu_x^o) = 0 \quad (6.8)$$

como,  $s + s^o = 1$ , segue que:

$$s_t + s_t^o = (s + s^o)_t = 0 \quad (6.9)$$

e portanto:

$$(\nu + \nu^o)_x = 0 \quad (6.10)$$

Assim a taxa de escoamento volumétrico total

$$\nu^T = \nu + \nu^o \quad (6.11)$$

é constante. Eliminando a pressão nas relações de Darcy obtemos:

$$\nu = \nu^T \frac{\lambda}{\lambda + \lambda^o} = \nu^T . f \quad (6.12)$$

onde  $f = f(s, c)$  é a conhecida função de fluxo fracionário:

$$f = \frac{\lambda}{\lambda + \lambda^o} \quad (6.13)$$

Nesta função a dependência de  $s$  é inerente as permeabilidades relativas  $k$  e  $k^o$  enquanto que a dependência de  $c$  é introduzida através da viscosidade  $\mu = \mu(c)$  da fase aquosa.

Usando  $\nu^T$  e  $.f$ , o modelo se rescreve na forma:

$$\phi.s_t + \nu^T . f = 0 \quad (6.14)$$

$$\phi.(s.c)_t + \nu^T (c..f)_x = 0 \quad (6.15)$$

Na adimensionalização usamos o comprimento da região considerada que é denotado por  $L$  e outras variáveis já definidas. Introduzimos portanto um novo sistema de coordenadas. Aqui  $\phi$  é a porosidade da rocha que é tomada constante.



$$\begin{cases} x' = \frac{x}{L} \\ t' = \frac{v^2 t}{\phi \cdot L} \end{cases} \quad (6.16)$$

onde o sistema se rescreve na forma:

$$\begin{cases} s_t + f(s, c)_x = 0 \\ (s \cdot c)_t + (c \cdot f(s, c))_x = 0 \end{cases} \quad (6.17)$$

para o caso sem adsorção e

$$\begin{cases} s_t + f(s, c)_x = 0 \\ (s \cdot c + a(c))_t + (c \cdot f(s, c))_x = 0 \end{cases} \quad (6.18)$$

para o caso com adsorção, onde pôr simplicidade, usaremos  $(x, t)$  para denotar as variáveis adimensionalizadas. Para maiores detalhes ver [JOH] ou [ISA].

## 6.2. Solução fraca

Consideremos o problema de Riemann para um sistema de equações da forma:

$$u_t + F(u)_x = 0 \quad (6.19)$$

com condições iniciais:

$$u(x, 0) = \begin{cases} u_e & \text{se } x < 0 \\ u_d & \text{se } x > 0 \end{cases} \quad (6.20)$$

Em geral as soluções de problemas do tipo (6.19)-(6.20) são descontínuas. Assim nós procuramos solução no sentido fraco, ou seja,  $u(x, t)$  é solução se satisfaz:

$$\int_0^\infty \int_{-\infty}^\infty (u(x, t) \cdot \varphi_t(x, t) + F(u) \cdot \varphi_x(x, t)) dx \cdot dt + \int_{-\infty}^\infty u(x, 0) \cdot \varphi(x, 0) dx \quad (6.21)$$

para toda função  $\varphi$  com suporte compacto.

## 6.3. Ondas simples

As ondas simples são soluções no sentido fraco e podem ser descontínuas, chamadas de ondas de choque e as soluções contínuas chamadas de ondas de rarefação.

### A) Ondas de rarefação

São soluções contínuas do problema (6.19)-(6.20) da forma :

$$u(x, t) = \begin{cases} u_e & \text{se } \xi < \lambda(u_e) \\ v(\xi) & \text{se } \xi = \lambda(v) \\ u_d & \text{se } \xi > \lambda(u_d) \end{cases} \quad (6.22)$$

onde  $\xi = \frac{x}{t}$  e  $\lambda$  é um autovalor associado a um autovetor  $\vec{e}$ , e  $v$  é a curva integral do campo vetorial que conecta os estados  $u_e$  à  $u_d$  juntamente com propriedade que  $v$  seja crescente de  $u_e$  para  $u_d$

### B) Ondas de choque

Uma onda de choque conectando dois estados  $u_e = (s_e, c_e)$  e  $u_d = (s_d, c_d)$  é uma solução fraca do problema de Riemann (6.19)-(6.20) da forma:

$$u(x, t) = \begin{cases} u_e & \text{se } \xi < \delta \\ u_d & \text{se } \xi > \delta \end{cases} \quad (6.23)$$

onde  $\delta$  é a velocidade do choque. Para dar significado físico a estas soluções,  $u(x, t)$  deve satisfazer uma condição chamada de "Condição de Entropia" (ver [LEV],[JOH],[LAX]). Esta condição requer que as ondas de choque sejam evolucionárias, isto é, qualquer onda de choque deve ser limite de soluções de ondas associadas a sistemas de viscosidades.

A solução fraca da forma (6.23) deve satisfazer a condição de Rankine-Hugoniot (ver [LEV],[JOH],[ISA]) aplicadas as equações do sistema. Elas definem como deve ser as velocidades dos choques que formam a solução. Para o modelo de Isaacson, conclui-se que os choques devem ter velocidades dadas por:

$$\frac{f(s_d, c_d)}{s_d} = \frac{f(s_e, c_e)}{s_e} = \delta \quad (6.24)$$

se  $c_e \neq c_d$ . Os choques com estas velocidades serão chamados de c-choques.

Se  $c_e = c_d$ , os choques são os que aparecem na solução da equação de Buckley-Leverett e serão chamados de s-choques. Suas velocidades são dadas por:

$$f(s_d, c_e) - f(s_e, c_e) = \delta(s_d - s_e) \quad (6.25)$$

Assim (6.24) e (6.25) definem explicitamente a velocidade de choque  $\delta$ , para um c-choque e para um s-choque respectivamente para o caso sem adsorção.

Para o modelo com adsorção a condição de Rankine-Hugoniot fica nos dá:

$$\frac{f(s_d, c_d)}{s_d + h_{LR}} = \frac{f(s_e, c_e)}{s_e + h_{LR}} = \delta \quad (6.26)$$

como a velocidade dos c-choques ( ver [JOH]). Os s-choques terão velocidades dadas também por (6.25).

## 7. APÊNDICE B

Apresentamos aqui a documentação dos programas desenvolvidos para a simulação dos processos de recuperação citados nos capítulos 3 e 4.

Nossos programas foram todos desenvolvidos no ambiente MATLAB que possui uma linguagem de programação própria. Em vários momentos utilizamos rotinas pertencentes ao MATLAB, principalmente as rotinas que trabalham a parte de interpolação e a parte de geração de gráficos. Apresentamos também as listagens dos programas desenvolvidos.

Vamos descrever os quatro programas principais associados aos quatro casos estudados que são:

1. Solução do Problema de Riemann associado à injeção de água com polímeros sem adsorção ( Isaacson ).
2. Solução do Problema de Riemann associado à injeção de água com polímeros com adsorção ( Johansen & Whinter ).
3. Solução do Problema da injeção de bancos de água e água com polímeros - Caso sem adsorção.
4. Solução do Problema da injeção de bancos de água e água com polímeros - Caso com adsorção.

### 7.1. O Problema de Riemann - Modelo sem adsorção.

Utilizando a solução analítica do problema de Riemann apresentada por Isaacson ([ISA]) desenvolvemos um programa computacional para a simulação dos perfis de solução para todas as possíveis situações e um outro programa que calcula a produção de óleo pelo processo.

O programa consiste de quatro rotinas principais para o cálculo da solução semi-analítica que são: dados.m, buck.m, isacmtd.m e prodisac.m. Além disso

temos a rotina upwisac.m que calcula a solução utilizando o esquema numérico proposto no capítulo 4. Estas rotinas estão contidas no diretório a:\isaacson.

### 7.1.1. Dados.m

#### Objetivos:

Entrada de dados para o programa principal e cálculo de outros dados essenciais.

#### Dados de entrada:

SL = saturação à esquerda

SR = saturação à direita

CL = concentração à esquerda

CR = concentração à direita

swi = saturação de água inicial

sor = saturação de óleo residual

t = tempo

#### Dados de saída:

s = vetor saturação

c = vetor concentração

fl = fluxo fracionário associado à CL

fr = fluxo fracionário associado à CR

dfl = derivada de fl

dfr = derivada de fr

### 7.1.2. Buck.M

#### Objetivos:

Calcular a solução para a equação de Buckley-Leverett

#### Dados de entrada:

x = malha na direção x

s = vetor saturação

cw = parâmetro para a função de fluxo fracionário

f = função de fluxo fracionário

df = derivada da função f

sd = saturação à direita

se = saturação à esquerda

t = tempo

#### Algoritmo:

Realizado de acordo com a solução apresentada em [ISA].

**Dados de saída:**

satura : vetor solução dado em termos da saturação

### 7.1.3. Isacmtd.m

**Objetivos:**

Calcula a saturação  $s(x, t)$  e a concentração  $c(x, t)$  para valores de  $t$ .

**Dados de entrada:**

$x$  = malha na direção  $x$

SL = valor da saturação à esquerda

SR = valor da saturação à direita.

CL = valor da concentração à esquerda.

CR = valor da concentração à direita.

$t$  = tempo

**Algoritmo:**

Realizado de acordo com apresentado no capítulo 2. É composto por seis subrotinas: casol1.m, casol2.m, casol3.m, casor1.m, casor2.m e casor3.m

**Dados de saída:**

satu = saturação  $s(x, t)$

conc = concentração  $c(x, t)$

### 7.1.4. Prodisac.m

**Objetivos:**

Calculo da evolução da produção de óleo com o tempo.

**Dados de entrada:**

$x$  = malha na direção  $x$

tdmax = tempo adimensional

**Algoritmo:**

Resolvemos numericamente a integral dada no capítulo 4 que mede a produção de óleo com o tempo, utilizando a rotina TRAPZ.M que é uma rotinas do pacote MATLAB.

**Dados de saída:**

$A(i,j) = s(x(i), tempo(j))$ ..

$B(i,j) = c(x(i), tempo(j))$ .

Produção = produção no instante  $TM(j)$ .

### 7.1.5. Upwisac.m

#### Objetivos:

Implementar o esquema upwind para o sistema de equações.

#### Dados de entrada:

$dx$  = incremento na direção  $x$

$t$  = tempo para o cálculo

$SL$  = saturação a esquerda

$SR$  = saturação à direita

$CL$  = concentração à esquerda.

$CR$  = concentração à direita.

#### Algoritmo:

Dado pelo esquema numérico convergente apresentado no capítulo 4.

#### Dados de saída:

$S1$  = vetor saturação

$C1$  = vetor concentração

## 7.2. O Problema de Riemann - Modelo com adsorção.

Utilizando a solução analítica apresentada no capítulo 2 para o caso da injeção de água com polímeros considerando o efeito da adsorção, elaboramos um programa para que calcula a solução quando  $CL > CR$ .

Para obtermos a solução  $s(x, t)$  e  $c(x, t)$  são utilizadas três rotinas principais que são: dados.m, joha.m e produjoh.m. Para a solução numérica dada pelo esquema numérico do tipo upwind temos a rotina upwjoha.m. Estas rotinas estão contidas no diretório a:\johansen.

Passemos então a descreve-los um a um.

### 7.2.1. Dados.m

A rotina dados.m que utilizamos é a mesma dada em 1.

### 7.2.2. Joha.m

#### Objetivos:

Calcula a solução para o problema da injeção de água com polímeros associado ao sistema de equações com o termo adsorção.

#### Dados de entrada:

x = malha na direção x  
SL = valor da saturação à esquerda.  
SR = valor da saturação à direita  
CL = valor da concentração à esquerda  
CR = valor da concentração à direita.  
sstar = estado intermediário  $s^*$  da literatura calculado pela rotina intstar.m  
sl = estado intermediário calculado pela rotina inter1.m  
sk = estado intermediário calculado pela rotina interk.m  
s = vetor saturação inicial  
fl = função de fluxo fracionário  $f(s, CL)$   
fr = função de fluxo fracionário  $f(s, CR)$   
dfl = derivada da função fl  
dfr = derivada da função fr  
t = tempo

**Algoritmo:**

O algoritmo é construído de acordo com a solução apresentada em [JOH].

**Dados de saída :**

satu = vetor saturação que dá a solução  $s(x(i), t)$   
conc = vetor concentração que dá a solução  $c(x(i), t)$ .

### 7.2.3. Prodjoh.m

**Objetivos:**

Calculo da evolução da produção de óleo com o tempo.

**Dados de entrada:**

x = malha na direção x  
tdmax = tempo adimensional  
TM = vetor tempo

**Algoritmo:**

Resolvemos numericamente a integral apresentada no capítulo 4 que calcula a produção de óleo com o tempo, utilizando a rotina TRAPZ.M do pacote MATLAB.

**Dados de saída:**

A(i,j) = valor da saturação  $s(x(i), tempo(j))$ ..  
B(i,j) = valor da concentração  $c(x(i), tempo(j))$ ..  
Produção = produção quando  $t = tdmax$



Outras quatro rotinas calculam os estados intermediários que aparecem na construção da solução. São elas, Intstar.m, Inter1.m, Interk.m e Rotst.m.

#### 7.2.4. Intstar.m

##### Objetivos:

Calcular o estado intermediário  $s^*$  que é o valor que satisfaz a equação:  $\frac{\partial f}{\partial s}(s, CL) = \frac{f(s, c)}{s + h_{LR}}$  na forma adimensional dada no capítulo 3.

##### Dados de entrada:

- s = vetor saturação
- CR = valor da concentração  $c(x, 0)$
- CL = valor da concentração  $c(0, t)$
- swi = valor da saturação de água inicial.
- sor = valor da saturação de óleo residual.
- fl = vetor  $f(s, CL)$
- fr = vetor  $f(s, CR)$
- n = número de coordenadas do vetor s.

##### Algoritmo:

Resolvemos utilizando o método da bissecção.

##### Dados de saída:

- sstar = estado intermediário  $s^*$

#### 7.2.5. Inter1.m

##### Objetivos:

Calcula o valor intermediário  $s1$  que satisfaz a equação  $\frac{f(s^*, CL)}{s^* + h_{LR}} = \frac{f(s1, CR)}{s1 + h_{LR}}$

com a seguinte condição,  $\frac{\partial f}{\partial s}(s1, CR) > \frac{f(s1, CR)}{s1 + h_{LR}}$

##### Dados de entrada:

- s = vetor saturação
- CR = valor da concentração  $c(x, 0)$
- CL = valor da concentração  $c(0, t)$
- swi = valor da saturação de água inicial
- sor = valor da saturação de óleo residual
- fl = vetor  $f(s, CL)$
- fr = vetor  $f(s, CR)$

n = número de coordenadas do vetor s  
sstar= valor do estado intermediário s\*

**Algoritmo:**

Resolvemos utilizando o método da bisseção.

**Dados de saída:**

s1 =: estado intermediário s1

### 7.2.6. Interk.m

**Objetivos:**

Calcula o valor intermediário sk que satisfaz a equação  
$$\frac{f(s^*, c_L)}{s^* + h_{LR}} = \frac{f(sk, c_R)}{sk + h_{LR}}$$
 com a seguinte condição  $\frac{\partial f}{\partial s}(x, c_R) < \frac{f(x, c_R)}{x + h_{LR}}$ , onde s\*  
é calculado em Intstar.m

**Dados de entrada:**

s = vetor saturação  
CR = valor da concentração  $c(x, 0)$   
CL = valor da concentração  $c(0, t)$   
swi = valor da saturação de água inicial  
sor = valor da saturação de óleo residual  
fl = vetor  $f(s, CL)$   
fr = vetor  $f(s, CR)$   
n = número de coordenadas do vetor s  
sstar= valor do estado intermediário s\*

**Algoritmo:**

Resolvemos utilizamos o método da bisseção.

**Dados de saída:**

sk = estado intermediário sk

### 7.2.7. Rotst.m

**Objetivos:**

Resolver a equação  $\lambda^S(x, c) = \lambda^C(x, c)$  para cada valor de c intervalo [0, 1],  
dando origem a curva de transição T

**Dados de entrada:**

s = vetor saturação  
c = vetor concentração  
df = derivada da função de fluxo fracionário

swi= saturação de água inicial  
sor= saturação de óleo residual

**Algoritmo:**

Resolvemos utilizando o método da bisseção.

**Dados de saída:**

stc = vetor que depende de c .

### 7.2.8. Upwjoha.m

**Objetivos:**

Implementar o esquema upwind para o sistema de equações que modela o problema.

**Dados de entrada:**

dx = incremento na direção x

t = tempo para o cálculo

SL= saturação a esquerda

SR= saturação à direita

CL= concentração à esquerda.

CR= concentração à direita.

**Algoritmo:**

Dado pelo esquema numérico convergente apresentado no capítulo 4.

**Dados de saída:**

S1 = vetor saturação

C1 = vetor concentração

### 7.3. O Problema da Injeção de Bancos - Caso sem adsorção.

Como vimos no capítulo 3 para explicitarmos a solução para o caso da injeção de dois ou mais bancos de água e de água com polímeros devemos inicialmente construir as curvas de descontinuidade associadas. O algoritmo principal portanto consiste em construir estas curvas. Para o cálculo da solução, dividimos o nosso trabalho em três rotinas principais que são: dados.m, durvalg.m e probanc.m . A rotina dados.m é uma rotina de entrada e cálculo de dados, a rotina curvalg.m composta de três subrotinas curvay1.m, curvay2.m e curvay3.m que calculam as curvas de descontinuidades do problema. Por final a rotina Prodbanc.m que além de explicitar a solução para um valor de t fixado ( perfil ) também calcula a evolução da produção com o passar do tempo.

Para encontrarmos o valor da produção de óleo devemos rodar as rotinas `Dados.m`, `Curvalg.m` e `Prodbanc.m` nesta ordem. Alguns valores podem ser alterados na execução do programa e são eles: `cinj2`, `T0`, `T1`, `swi` e `sor`. Outra rotina importante é a rotina `Upvalg.m` que calcula a solução numérica, utilizando o esquema numérico do tipo `upwind` dado no capítulo 4. Estas rotinas estão contidas no diretório `a:\isacbanc`.

Passamos então a descrever cada uma destas rotinas citadas acima.

### 7.3.1. `Dados.m`

#### **Objetivos:**

Manipular dados essenciais ao problema. Calcula estados intermediários que aparecem na solução do problema de Riemann, através das subrotinas `intstar.m`, `interl.m` e `interk.m` que já foram apresentadas anteriormente.

#### **Dados de entrada:**

Entre com valores de :

`T0`, `T1`, `so`, `co`, `sinj1`, `sinj2`, `sinj3`, `cinj1`, `sinj2`, `sinj3`, `swi`, `sor`, `n`, `dx`, `dt`.

#### **Algoritmo:**

È uma rotina de entrada e manipulação de dados

#### **Dados de saída:**

Vetores : `s`, `f`, `fr`, `x`

Estados intermediários: `s1`, `sk`, `sm` e `sstar`.

### 7.3.2. `Curvalg.m`

#### **Objetivos:**

Calcular as curvas de descontinuidades da solução

#### **Dados de entrada:**

`T0`, `T1`, valor da rarefação do primeiro leque.

#### **Algoritmo:**

Composto pelas três subrotinas `curvay1.m`, `curvay2.m` e `curvay3.m`

#### **Dados de saída:**

`y1`, `y2`, `y3`, `y4`, `y5`, `yo`, `to`, `t1`, `t2`, `t3`, `t4`, `t5`

### 7.3.3. `Curvay1.m`

#### **Objetivos:**

Calcular a curva de descontinuidade  $y_1$  definida no capítulo 3.

**Dados de entrada:**

s = vetor saturação

nt = número natural

sk = estado intermediário que aparece na solução

sstar = estado intermediário que aparece na solução.

swi = saturação de água inicial

sor = saturação de óleo residual

T0 = tamanho do primeiro banco

**Algoritmo:**

Dado no capítulo 3 que trata da solução do problema da injeção de bancos.

**Dados de saída:**

y1 = ordenada da curva y1

t1 = abcissa da curva y1

**7.3.4. Curvay2.m****Objetivos:**

Calcular a curva de descontinuidade  $y_2$  definida no capítulo 3.

**Dados de entrada:**

s = vetor saturação

nt = número natural

s1 = estado intermediário que aparece na solução

sk = estado intermediário que aparece na solução

sstar = estado intermediário que aparece na solução.

swi = saturação de água inicial

sor = saturação de óleo residual

T0 = tamanho do primeiro banco

**Algoritmo:**

Dado no capítulo 3, que trata da solução do problema da injeção de bancos.

**Dados de saída:**

y2 =: ordenada da curva y2

t2 = abcissa da curva y2

**7.3.5. Curvay3.m****Objetivos:**

Calcular a curva de descontinuidade  $y_3$  definida no capítulo 3.

**Dados de entrada:**

s = vetor saturação  
nt = número natural  
s1 = estado intermediário que aparece na solução  
sk = estado intermediário que aparece na solução  
sstar = estado intermediário que aparece na solução.  
swi = saturação de água inicial  
sor = saturação de óleo residual  
T0 = tamanho do primeiro banco  
T1 = tamanho do segundo banco  
y1 = ordenada da curva y1  
t1 = abcissa da curva y1

**Algoritmo:**

Dado no capítulo 3, que trata da solução do problema da injeção de bancos.

**Dados de saída:**

y3 : ordenada da curva y3  
t3 : abcissa da curva y3

### 7.3.6. Prodbanc.m

**Objetivos:**

Calcular o valor da produção de óleo em função do tempo  $t$ .

**Dados de entrada:**

tdmax = tempo máximo relativo ao ultimo perfil de saturação a ser calculado.  
x = malha na direção x  
T0 = tamanho do primeiro banco  
T1 = tamanho do segundo banco  
y1 = ordenada da curva y1  
t1 = abcissa da curva y1  
y2 = ordenada da curva y2  
t2 = abcissa da curva y2  
y3 = ordenada da curva y3  
t3 = abcissa da curva y3  
sinj1= saturação de injeção de água no primeiro banco  
sinj2= saturação de injeção de água no segundo banco  
sinj3= saturação de injeção de água no terceiro banco  
cinj1= concentração de injeção de polímero no primeiro banco  
cinj2= concentração de injeção de polímero no segundo banco

cinj1= concentração de injeção de polímero no terceiro banco

**Algoritmo:**

Calculamos o valor da saturação e da concentração em  $x = 1$  e em seguida utilizamos a regra do trapézio para resolver a integral apresentada no capítulo 4 que calcula a produção de óleo.

**Dados de saída:**

$$A(i, j) = s(x(i), tempo(j)).$$

$$B(i, j) = c(x(i), tempo(j)).$$

Produção: valor da produção de óleo quando  $t = t_{dmax}$ .

### 7.3.7. Upwalg.m

**Objetivos:**

Implementar o esquema upwind para o sistema de equações que modela o problema.

**Dados de entrada:**

dx = incremento na direção x

t = tempo para o cálculo

T0 = tempo em começamos a injeção do segundo banco.

T1 = tempo em que começamos a injeção do terceiro banco.

sinj1= saturação de injeção no primeiro banco

sinj2= saturação de injeção no segundo banco

sinj3= saturação de injeção no terceiro banco

cinj1= concentração no primeiro banco.

cinj2= concentração no segundo banco.

cinj3= concentração no terceiro banco.

**Algoritmo:**

Dado pelo esquema numérico convergente apresentado no capítulo 4.

**Dados de saída:**

S1 = vetor saturação

C1 = vetor concentração

## 7.4. O Problema da Injeção de Bancos - com adsorção.

Como observamos no capítulo anterior o caso da injeção de bancos com adsorção foi desenvolvido apenas para dois bancos. Sendo o primeiro de água seguido de um outro de água com polímero. Neste caso são quatro as rotinas a saber: dados.m,

curvas.m, twobanc.m e prodads.m que devem ser rodados nesta ordem. A rotina upwads.m implementa o método numérico (tipo upwind). Estas rotinas estão contidas no diretório a:\johabanc.

#### 7.4.1. Dados.m

Esta rotina é análoga à dada no parágrafo anterior.

#### 7.4.2. Curvas.m

##### Objetivos:

Calcular as descontinuidades do problema.

##### Dados de entrada:

Todos os dados obtidos e manipulados na rotina Dados.m

##### Algoritmo:

O descrito no capítulo 3 .

##### Dados de saída:

y11, y12, y13, y14, y15, t11, t12, t13, t14, t15

#### 7.4.3. Twobanc.m

##### Objetivos:

Calcula os perfis de saturação  $s(x, t)$  e concentração  $c(x, t)$

##### Dados de entrada:

Obtidos nas duas rotinas anteriores.

##### Algoritmo:

Através do cálculo das descontinuidades e das características calculamos os vetor solução.

##### Dados de saída:

$A(i, j) = s(x(i), tempo(j))..$

$B(i, j) = c(x(i), tempo(j)).$

#### 7.4.4. Prodads.m

##### Objetivos:

Calcular a produção no tempo t dado.

##### Dados de entrada:

Obtidos nas duas rotinas anteriores.



**Algoritmo:**

Resolvemos numericamente a integral apresentada no capítulo 4 que mede a produção de óleo com o tempo, utilizando a rotina TRAPZ.M do pacote MATLAB.

**Dados de saída:**

prodador: produção no tempo t.

**7.4.5. Upwads.m****Objetivos:**

Implementar o esquema upwind para o sistema de equações que modela o problema.

**Dados de entrada:**

dx = incremento na direção x

t = tempo para o cálculo

T0 = tempo em começamos a injeção do segundo banco.

T1 = tempo em que começamos a injeção do terceiro banco.

sinj1= saturação de injeção no primeiro banco.

sinj2= saturação de injeção no segundo banco.

sinj3= saturação de injeção no terceiro banco.

cinj1= concentração no primeiro banco.

cinj2= concentração no segundo banco.

cinj3= concentração no terceiro banco.

**Algoritmo:**

Dado pelo esquema numérico convergente apresentado no capítulo 4

**Dados de saída:**

S1 = vetor saturação

C1 = vetor concentração

## BIBLIOGRAFIA

- [BED] BEDRIKOVETSKY, P.- Mathematical Theory of Oil and Gas Recovery - Kluwer Academic Publishers - 1993.
- [BRB] BARENBLATT, G.I. E V.M. ENTOV, V.M. RYZHIK - Theory of fluid flows through natural rocks - Kluwer Academic Publishers - 1990.
- [CRA] CRANDALL, M.G. & A. MAJDA - Monotone difference approximations for scalar conservation laws. - Mathematics of Computation - 1980 - Vol. 24 - n° 149 - pp 1-21.
- [ISA] ISAACSON, E.L. - Global solution of a Riemann Problem for a non strictly hyperbolic system of conservation laws arising in enhanced oil recovery -1980 - Rockefeller University preprints .
- [JOH] JOHANSEN, T. & R. WINTHER - The solution of Riemann Problem for a hyperbolic system of conservation laws modeling polymer flooding - SIAM - J.Math.Anal.,19, 1988 - pp. 541 - 566.
- [LAK] LAKE, L.W. - Enhanced oil recovery - Prentice Hall - New Jersey - 1989.
- [LAX] LAX, P.D.- Hyperbolic systems of conservation laws and the mathematical theory of shock waves -1973- SIAM Regional Conference Series in Applied Mathematics.
- [LEV] LE VEQUE, R.J. - Numerical methods for conservation laws - Birkhäuser Verlag-1990 .
- [MAR] MARLE, CHARLES M. - Multiphase Flow in Porous Media - 1981 - Editions Technip, Paris.
- [ODE] ODEH, AZIZ S. - On overview of mathematical modeling of behavior of hydrocarbon reservoirs - SIAM Review - 1982 - Vol. 24 - pp 265-273.

- [TVT] TVEITO, A. & R, WINTHER - Convergence of a non conservative finite difference scheme for a system of hyperbolic conservation laws - Dif. Integral Equations - Vol. 3 - n° 5 - 1990 - pp. 979-1000.
- [WSH] WALSH, M. & L.W.LAKE - Applying fractional flow theory to solvent flooding and chase fluids - Journal of Petroleum Science and Engineering - 2 -1989 - pp 281-303.

## Listagem dos programas computacionais

### 1 - Rotina Buck.m

```
%Buck.m
%Esta rotina calcula o perfil da Solução da equação de Buckley - Leverett
f = fract(s,cw);
df = derfract(s,cw);
fd = interp1(s,f,sd);
[maxi,i1]=max(df);
sinflex=s(i1);
if se > sd
    if sd >= sinflex
        clear a1;clear a2;
        clear sra; clear xra;
        vi=interp1(s,df,se);
        vf=interp1(s,df,sd);
        sra=se:(sd-se)/nt:sd;
        xra=interp1(s,df,sra)*t;
        for i=1:nx;
            if x(i) < vi*t
                satura(i)= se;
            elseif x(i) >= vi*t & x(i) < vf*t
                satura(i)=interp1(xra,sra,x(i));
            else
                satura(i)=sd;
            end;
        end;
    else
        intersm;
        if se > sm
            clear a1;clear a2;
            clear sra; clear xra;
            sra=se:(sm-se)/nt:sm;
            xra=interp1(s,df,sra)*t;
            vi=interp1(s,df,se) ;
            vf=interp1(s,df,sm) ;
            for i=1:nx;
                if x(i) < vi*t
                    satura(i)= se;
```

```

        elseif x(i) >= vi*t & x(i) < vf*t
            satura(i)=interp1(xra,sra,x(i));
        else
            satura(i)=sd;
        end;
    end;
else
    fse=interp1(s,f,se);
    fsd=interp1(s,f,sd);
    vi=(fse-fsd)/(se-sd);
    vf=(fse-fsd)/(se-sd);

    for i=1:nx;
        if x(i) < vi*t;
            satura(i)= se;
        else
            satura(i)= sd;
        end;
    end;
end;
end;
else
if sd <= sinflex ;
    clear a1;clear a2;
    clear sra; clear xra;
    vi=interp1(s,df,se);
    vf=interp1(s,df,sd);
    sra=se:(sd-se)/nt:sd;
    xra=interp1(s,df,sra)*t;

    for i=1:nx;
        if x(i) < vi*t
            satura(i)= se;
        elseif x(i) >= vi*t & x(i) < vf*t
            satura(i)=interp1(xra,sra,x(i));
        else
            satura(i)=sd;
        end;
    end;
end;
end;

```

```

else
    intersm;
    if se < sm
        clear a1;clear a2;
        clear sra; clear xra;
        sra=se:(sm-se)/nt:sm;
        xra=interp1(s,df,sra)*t;
        vi=interp1(s,df,se);
        vf=interp1(s,df,sd);

        for i=1:nx;
            if x(i) < vi*t
                satura(i)= se;
            elseif x(i) >= vi*t & x(i) < vf*t
                satura(i)=interp1(xra,sra,x(i));
            else
                satura(i)=sd;
            end;
        end;
    end;
else
    fse=interp1(s,f,se);
    fsd=interp1(s,f,sd);
    vi=(fse-fsd)/(se-sd);
    vf=(fse-fsd)/(se-sd);
    for i=1:nx;
        if x(i) < vi*t;
            satura(i)= se;
        else
            satura(i)= sd;
        end;
    end;
end;
end;
end;

```

## 2 - Rotina casol1.m

```
% UR pertence a L1
% Cálculo do estado intermediário s1
if CR < CL
    clear SC;
    clear CS;
    SC(1)=SL;
    CS(1)=CL;
    DS=-0.001;
    i=1;
    while CS(i) > CR
        dc=dfc1(SC(i),CS(i));
        ls=derfract(SC(i),CS(i));
        lc=fract(SC(i),CS(i))./(SC(i)+swi/snom);
        SC(i+1)=SC(i)+DS;
        CS(i+1)=CS(i) + DS*( (lc-ls)./dc );
        i=i+1;
    end;
    CS(i)=CR;
    s1=(SC(i)+SC(i-1))/2;
else
    clear SC;
    clear CS;
    SC(1)=SL;
    CS(1)=CL;
    DS=0.001;
    i=1;
    while CS(i) < CR
        dc=dfc1(SC(i),CS(i));
        ls=derfract(SC(i),CS(i));
        lc=fract(SC(i),CS(i))./(SC(i)+swi/snom);
        SC(i+1)=SC(i)+DS;
        CS(i+1)=CS(i) + DS*( (lc-ls)./dc );
        i=i+1;
    end;
    CS(i)=CR;
    s1=(SC(i)+SC(i-1))/2;
end;
% Descrevendo a solução
```

```

se=s1;
sd=SR;
cw=CR;
fd=interp1(s,fr,sd);
buck;
v2i=vi;v2f=vf;
v1i=lampc1(s1,CR,swi,sor);
v1f=v1i;

for i=1:nx;
    if x(i) < v1i*t
        satu(i)=SL;
        conc(i)=CL;
    elseif x(i) >= v1i*t & x(i) < v2i*t
        satu(i)=s1;
        conc(i)=CR;
    elseif x(i) >= v2i*t & x(i) < v2f*t
        satu(i)=satura(i);
        conc(i)=CR;
    else
        satu(i)=SR;
        conc(i)=CR;
    end;
end;
end;

```



### 3 - Rotina casol2.m

```
% UR pertence à L2
if CL < CR
    clear CS;
    clear SC;
    CS(1)=CR;
    SC(1)=SR;
    DS=0.001;
    i=1;
    while CS(i) > CL
        dc=dfc1(SC(i),CS(i));
        ls=derfract(SC(i),CS(i));
        lc=fract(SC(i),CS(i))./(SC(i)+swi/snom);
        SC(i+1)=SC(i)+DS;
        CS(i+1)=CS(i) + DS*( (lc-ls)./dc );
        i=i+1;
    end;
    CS(i)=CL;
    s1=interp1(CS,SC,CL);
else
    clear CS;
    clear SC;
    CS(1)=CR;
    SC(1)=SR;
    DS=-0.001;
    i=1;
    while CS(i) < CL
        dc=dfc1(SC(i),CS(i));
        ls=derfract(SC(i),CS(i));
        lc=fract(SC(i),CS(i))./(SC(i)+swi/snom);
        SC(i+1)=SC(i)+DS;
        CS(i+1)=CS(i) + DS*( (lc-ls)./dc );
        i=i+1;
    end;

    CS(i)=CL;
    s1=interp1(CS,SC,CL);
end;
```

```

% Descrevendo a solução
se=SL;
sd=s1;
cw=CL;
fd=interp1(s,fl,sd);
buck;
v1=vi;
v2=vf;
v3=lampc1(s1,CL,swi,sor);

for i=1:nx;
    if x(i) < v1*t
        satu(i)=SL;
        conc(i)=CL;
    elseif x(i) >= v1*t & x(i) < v2*t
        satu(i)=satura(i);
        conc(i)=CL;
    elseif x(i) >= v2*t & x(i) < v3*t
        satu(i)=s1;
        conc(i)=CL;
    else
        satu(i)=SR;
        conc(i)=CR;
    end;
end;
end;

```

#### 4 - Rotina casol3.m

```
% UR=(SR,CR) pertence à L3
STR=interp1(c,st,CR);
s2=STR;
clear CS;
clear SC;
CS(1)=CR;
SC(1)=s2;
DS=0.001;
i=1;

while CS(i) > CL
    dc=dfc1(SC(i),CS(i));
    ls=derfract(SC(i),CS(i));
    lc=fract(SC(i),CS(i))./(SC(i)+swi/snom);
    SC(i+1)=SC(i)+DS;
    CS(i+1)=CS(i) + DS*( (lc-ls)./dc );
    i=i+1;
end;
CS(i)=CL;
s1=SC(i);

% Descrevendo a solução
se=SL;
sd=s1;
cw=CL;
fd=interp1(s,fl,sd);
buck;
satura1=satura;
vli=vi;
vlf=vf;

se=s2;
sd=SR;
cw=CR;
fd=interp1(s,fr,sd);
buck;
satura2=satura;
```

```

v3i=vi;
v3f=vf;
v2i=lampc1(s1,CL,swi,sor);
v2f=v2i;

for i=1:nx;
    if x(i) < v1i*t
        satu(i)=SL;
        conc(i)=CL;
    elseif x(i) >= v1i*t & x(i) < v2i*t
        satu(i)=satural(i);
        conc(i)=CL;
    elseif x(i) >= v2i*t & x(i) < v3i*t
        satu(i)=s1;
        conc(i)=CL;
    elseif x(i) >= v3i*t & x(i) < v3f*t
        satu(i)=satura2(i);
        conc(i)=CR;
    else
        satu(i)=SR;
        conc(i)=CR;
    end;
end;

```

## 5 - Rotina casor1.m

% Caso em UR pertence a R1

% cálculo do estado intermediário

```
clear SC;
clear CS;
SC(1)=STL;
CS(1)=CL;
DS=-0.001;
i=1;
```

```
while CS(i) > CR
    dc=dfc1(SC(i),CS(i));
    ls=derfract(SC(i),CS(i));
    lc=fract(SC(i),CS(i))./(SC(i)+swi/snom);
    SC(i+1)=SC(i)+DS;
    CS(i+1)=CS(i) + DS*( (lc-ls)./dc );
    i=i+1;
end;
```

```
CS(i)=CR;
s2=(SC(i)+SC(i-1))/2;
```

% fim do cálculo intermediário s2

%Descrevendo a solução

```
se=SL;
sd=STL;
cw=CL;
fd=interp1(s,fl,sd);
buck;
satural=satura;
vli=vi;vlf=vf;
```

```
se=s2;
sd=SR;
cw=CR;
fd=interp1(s,fr,sd);
```

```

buck;
satura2=satura;
v3i=vi;
v3f=vf;
v2i=v1f;
v2f=v2i;

for i=1:nx;
    if x(i) < v1i*t
        satu(i)=SL;
        conc(i)=CL;
    elseif x(i) >= v1i*t & x(i) < v2i*t
        satu(i)=satural(i);
        conc(i)=CL;
    elseif x(i) >= v2i*t & x(i) < v3i*t
        satu(i)=s2;
        conc(i)=CR;
    elseif x(i) >= v3i*t & x(i) < v3f*t
        satu(i)=satura2(i);
        conc(i)=CR;
    else
        satu(i)=SR;
        conc(i)=CR;
    end;
end;
end;

```

## 6- Rotina casor2.m

```
% Se UR=(SR,CR) pertence à R2

if CR > CL
    clear CS;
    clear SC;
    CS(1)=CR;
    SC(1)=SR;
    DS=0.001;
    i=1;
    while CS(i) > CL
        dc=dfc1(SC(i),CS(i));
        ls=derfract(SC(i),CS(i));
        lc=fract(SC(i),CS(i))./(SC(i)+swi/snom);
        SC(i+1)=SC(i)+DS;
        CS(i+1)=CS(i) + DS*( (lc-ls)./dc );
        i=i+1;
    end;
    CS(i)=CL;
    s1=(SC(i)+SC(i-1))/2;
else
    clear CS;
    clear SC;
    CS(1)=CR;
    SC(1)=SR;
    DS=-0.001;
    i=1;
    while CS(i) < CL
        dc=dfc1(SC(i),CS(i));
        ls=derfract(SC(i),CS(i));
        lc=fract(SC(i),CS(i))./(SC(i)+swi/snom);
        SC(i+1)=SC(i)+DS;
        CS(i+1)=CS(i) + DS*( (lc-ls)./dc );
        i=i+1;
    end;
    CS(i)=CL;
    s1=(SC(i)+SC(i-1))/2;
end;
```

```

%Calculo do perfil s(x,t) e c(x,t)
se=SL;
sd=s1;
cw=CL;
fd=interp1(s,fl,sd);
buck;
v1i=vi;
v1f=vf;
v2i=lampr1(s1,CL,swi,sor);
v2f=v2i;

for i=1:nx;
    if x(i) < v1i*t
        satu(i)=SL;
        conc(i)=CL;
    elseif x(i) >= v1i*t & x(i) < v1f*t
        satu(i)=satura(i);
        conc(i)=CL;
    elseif x(i) >= v1f*t & x(i) < v2i*t
        satu(i)=s1;
        conc(i)=CL;
    else
        satu(i)=SR;
        conc(i)=CR;
    end;
end;
end;

```



## 7 - Rotina casor3.m

```
% UR=(SR,CR) pertence à L3

s2=STR;
clear CS;
clear SC;
CS(1)=CR;
SC(1)=s2;
DS=0.001;
i=1;

while CS(i) > CL
    dc=dfc1(SC(i),CS(i));
    ls=derfract(SC(i),CS(i));
    lc=fract(SC(i),CS(i))./(SC(i)+swi/snom);
    SC(i+1)=SC(i)+DS;
    CS(i+1)=CS(i) + DS*( (lc-ls)./dc );
    i=i+1;
end;

CS(i)=CL;
s1=(SC(i)+SC(i-1))/2;

% Cálculo do perfil s(x,t) e c(x,t)

se=SL;
sd=s1;
cw=CL;
fd=interp1(s,fl,sd);
buck;
satural=satura;
vli=vi;
vlf=vf;

se=s2;
sd=SR;
cw=CR;
fd=interp1(s,fr,sd);
```

```

buck;
satura2=satura;
v3i=vi;
v3f=vf;
v2i=lampcl(s1,CL,swi,sor);
v2f=v2i;

for i=1:nx;
    if x(i) < v1i*t
        satu(i)=SL;
        conc(i)=CL;
    elseif x(i) >= v1i*t & x(i) < v2i*t
        satu(i)=satural(i);
        conc(i)=CL;
    elseif x(i) >= v2i*t & x(i) < v3i*t
        satu(i)=s1;
        conc(i)=CL;
    elseif x(i) >= v3i*t & x(i) < v3f*t
        satu(i)=satura2(i);
        conc(i)=CR;
    else
        satu(i)=SR;
        conc(i)=CR;
    end;
end;
end;

```

## 8 - Rotina dados.m

% Manipula dados do problema e fixa alguns valores.

```
xmax=1.001;
```

```
dx=.001;
```

```
x=0:dx:xmax;
```

```
nx=size(x,2);
```

```
n=3000;
```

```
s=0:1/n:1;
```

```
c=0:1/200:1;nc=size(c,2);
```

```
SL=input('SL=');
```

```
SR=input('SR=');
```

```
CL=input('CL=');
```

```
CR=input('CR=');
```

```
swi=input('swi=');
```

```
sor=input('sor=');
```

```
t=input('t=');
```

```
snom=1-swi-sor;
```

```
for j=1:nc;
```

```
    yc=c(j);
```

```
    rotst1;
```

```
    st(j)=stc;
```

```
end;
```

```
fl=fract(s,CL);
```

```
dfl=derfract(s,CL);
```

```
fr=fract(s,CR);
```

```
dfr=derfract(s,CR);
```

```
nt=200;
```

## 9 - Rotina intersm.m

```
% Cálculo do estado intermediário sm, que é a abcissa  
% do ponto de tangência entre a reta que passa por (sd,f(sd,cw))  
% e o gráfico de f(s,cw).
```

```
if sd < sinflex  
    exp=2;  
else  
    exp=1;  
end;
```

```
r1=1;r2=n;  
k=r2-r1;
```

```
while k > 1;  
    r=round((r1+r2)/2);  
    G = ((-1)^(exp))* ( df(r) - (f(r)-fd)/(s(r)-sd) );  
    if G > 0 ;  
        r1=r;  
    else  
        r2=r;  
    end;  
    k= r2-r1;  
end;
```

```
sm=(s(r1)+s(r2))/2;  
fsm=interp1(s,f,sm);  
dfsm=interp1(s,df,sm);
```

## 10 - Rotina isacmtd.m

% Calcula a solução do Problema de Riemann - Isaacson

```
fl=fract(s,CL);
```

```
fr=fract(s,CR);
```

```
dfr=derfract(s,CR);
```

```
dfl=derfract(s,CL);
```

```
STL=interp1(c,st,CL);
```

```
clear satu;
```

```
clear conc;
```

```
clear satura;
```

```
clear s1;
```

```
clear s2;
```

```
if CL == CR
```

```
    se=SL;
```

```
    sd=SR;
```

```
    cw=CL;
```

```
    buck;
```

```
    satu=satura;
```

```
    for j=1:nx;
```

```
        conc(j)=CL;
```

```
    end;
```

```
else
```

```
    if SL < STL ;
```

```
        xl=CL;
```

```
        yl=SL;
```

```
        na=2;
```

```
        rar;
```

```
        CSL=CS;
```

```
        SCL=SC;
```

```
        lcr=lampc1(SR,CR,swi,sor);
```

```
        lcl=lampc1(SL,CL,swi,sor);
```

```
        STR=interp1(c,st,CR);
```

```
        if SR >= STR ;
```

```
            if lcr < lcl
```

```
                casol2;
```

```
            else
```

```

        casol1;
    end;
else
    if CR > max(CSL)
        casol3;
    else
        casol1;
    end;
end;
else
    x1=CL;
    y1=STL;
    na=2;
    rar;
    CSL=CS;
    SCL=SC;
    lcr=lampc1(SR,CR,swi,sor);
    lcl=lampc1(SL,CL,swi,sor);
    STR=interp1(c,st,CR);
    lctl=lampc1(STL,CL,swi,sor);

    if SR >= STR ;
        if lcr <= lctl
            casor2;
        else
            casor1;
        end;

    else
        if CR > CL
            casor3;
        else
            casor1;
        end;
    end;
end;
end;
end;

```

## 11 - Rotina prodbuck.m

```
% Cálculo da Produção de óleo  
% Entre com o valor de tdmax : tempo t em que se calcula o  
% perfil ou a produção.
```

```
clear A1  
if tdmax <= 1  
    jt=10;  
else  
    jt=fix(tdmax);  
end;
```

```
% jt refina a partição para o cálculo da integral que  
% define a produção.
```

```
tmax=tdmax/snom;  
ntmax=jt*10;  
dt=tmax/ntmax;
```

```
if tmax == 0;  
    producao=0;  
else  
    tempo=0:dt:tmax;  
    for j1=1:ntmax+1;  
        t=tempo(j1);  
        buck;  
        A1(j1,:)=satura;  
    end;  
    Fbuck=1-fract(A1(:,nx),cw);  
    producao=trapz(tempo,Fbuck);  
end;
```

## 12 - Rotina prodisac.m

```
% Cálculo da Produção de óleo para o caso da injeção de polímero  
% Entre com o valor de tdmax : tempo t em que se calcula o  
% perfil ou a produção.
```

```
clear A;clear B;
```

```
if tdmax < 1
```

```
    jt=10;
```

```
else
```

```
    jt=fix(tdmax);
```

```
end;
```

```
snom=1-swi-sor;
```

```
f1=fract(s,CL);
```

```
dfl=derfract(s,CL);
```

```
fr=fract(s,CR);
```

```
dfr=derfract(s,CR);
```

```
tmax=tdmax/snom;
```

```
ntmax=jt*100;
```

```
dt=tmax/ntmax;
```

```
if tmax == 0;
```

```
    producao=0;
```

```
else
```

```
    tempo=0:dt:tmax;
```

```
    for j1=1:ntmax+1;
```

```
        t=tempo(j1);
```

```
        isacmtd;
```

```
        A(j1,:)=satu;
```

```
        B(j1,:)=conc;
```

```
    end;
```

```
    Fisac=1-fract(A(:,nx),B(:,nx));
```

```
    producao=trapz(tempo,Fisac);
```

```
end;
```



### 13 - Rotina rotst1.m

```
% Cálculo da curva de transição T
% entre com o vetor yc e o vetor s.

df=derfract(s,yc);
r1=1;
r2=n;
k=r2-r1;
while k > 1;
    r=round((r1+r2)/2);
    G = df(r) - fract(s(r),yc)./(s(r)+swi/snom);
    if G > 0 ;
        r1=r;
    elseif G < 0;
        r2=r;
    else
        saida=s(r);
        break;
    end;
    k= r2-r1;
end;

stc= (s(r1)+s(r2))./2;
```

#### 14 - Rotina upwisac.m

% Cálculo da solução utilizando método de diferenças finitas  
% Utilizamos o método do tipo upwind

```
ds=.001;  
xmax=1.01;  
dx1=xmax/nxmax;
```

```
processo=input('caso queira modelo de Buckley=Leverett faça processo=0 , caso  
queira modelo de Isaacson faça processo=1 , processo=')
```

```
snom=1-swi-sor;  
cfl=12./25;  
Tf=t;  
dt=dx1.*cfl;  
Tf./dt;  
clear x1;clear S1;clear C1;  
x1=0:dx1:xmax;  
delta = dt./dx1;  
Tp1=0:dt:Tf;  
j=1;  
nTp1=size(Tp1,2),  
nx1=size(x1,2);
```

% Condição Inicial

```
for i=2:nx1;  
    S(i)=SR;  
    C(i)=CR;  
end;  
S(1)=SL;  
C(1)=CL;  
% Fim da condição inicial
```

```
if processo == 1
```

```
    for j=1:nTp1;j;  
        S1(1)=SL;  
        C1(1)=CL;  
        for i=2:nx1;  
            S1(i) = S(i) - delta*(fract(S(i),C(i))-fract(S(i-1),C(i-1)));  
            g = fract(S1(i),C(i)) ./ ( S1(i) + swi/snom );
```

```

        C1(i) = C(i) - delta.*g.*(C(i)-C(i-1));
    end;
    S=S1;C=C1;
end;
else
    for j=1:nTp1;j,
        S1(1)=SL;
        for i=2:nx1;
            S1(i) = S(i) - delta*(fract(S(i),cw)-fract(S(i-1),cw ));
        end;
        S=S1;C=C1;
    end;
end;
end;

```

## 15 - Rotina bancos.m

% Calcula a solução para o caso da injeção de dois ou três bancos.

% Devemos entrar com o valor de t

```
T01=t1(nt+1);
```

```
T02=max(t2);
```

```
t=t/snom;
```

```
if T1 < T02
```

```
    if t <= T0 ;
```

```
        clear xr1;
```

```
        xr1=qs1*t;
```

```
        ao=min(xr1);
```

```
        al=max(xr1);
```

```
        for i=1:nx;
```

```
            if x(i) >= ao & x(i) <= al;
```

```
                satu(i)=interp1(xr1,rar1,x(i));
```

```
                conc(i)=co;
```

```
            else
```

```
                satu(i)=so;
```

```
                conc(i)=co;
```

```
            end;
```

```
        end;
```

```
    elseif T0 < t & t <= T01
```

```
        clear xr1;
```

```
        clear xr2k;
```

```
        clear rar2k;
```

```
        xr1=qs1*t;
```

```
        y1t=interp1(t1,y1,t);
```

```
        i=1;
```

```
        while t >= t1(i);
```

```
            k=i;
```

```
            i=i+1;
```

```
        end;
```

```
        xr2k(1)=0;
```

```
        rar2k(1)=sinj2;
```

```
        for i=2:k;
```

```
            xr2k(i)=qs2(i)*(t-t1(i-1)) + y1(i-1);
```

```

        rar2k(i)=rar2(i);
    end;

    ao=min(xr2k);
    a1=interp1(y1,t1,t);
    a2=max(xr1);
    for i=1:nx;
        if x(i) <= ao
            satu(i)=sinj2;
            conc(i)=cinj2;
        elseif x(i) > ao & x(i) <= a1;
            satu(i)=interp1(xr2k,rar2k,x(i));
            conc(i)=cinj2;
        elseif x(i) > a1 & x(i) <= a2
            satu(i)=interp1(xr1,rar1,x(i));
            conc(i)=cinj1;
        else
            satu(i)=so;
            conc(i)=co;
        end;
    end;
elseif T01 < t & t <= T1
    clear xr1; clear xr2
    xr1=qs1*t;
    xr2(1)=0;
    for i=2:nt+1;
        xr2(i)=qs2(i)*(t-t1(i)) + y1(i);
    end;
    y2t=interp1(t2,y2,t);
    a0=min(xr2);
    a1=max(xr2);
    a2=y2t;
    a3=max(xr1);
    for i=1:nx;
        if x(i) <= a0
            satu(i)=sinj2;
            conc(i)=cinj2;
        elseif x(i) > a0 & x(i) <= a1
            satu(i)=interp1(xr2,rar2,x(i));

```

```

        conc(i)=cinj2;
    elseif x(i) > a1 & x(i) <= a2
        satu(i)=s1;
        conc(i)=cinj1;
    elseif x(i) > a2 & x(i) <= a3;
        satu(i)=interp1(xr1,rar1,x(i));
        conc(i)=cinj1;
    else
        satu(i)=so;
        conc(i)=co;
    end;
end;
elseif T1 < t & t <= T02
    clear xr1;clear xr2;
    clear xr3k;clear rar3k;
    xr2(1)=0;
    for i=2:nt+1;
        xr2(i)=qs2(i)*(t-t1(i-1)) + y1(i-1);
    end;

    xr1=qs1*t;
    y3t=interp1(t3,y3,t);
    i=1;
    while t >= t3(i);
        k=i;
        i=i+1;
    end;
    clear xr3k;
    clear rar3k;

    if k < size(t2,2)-1
        xr3k(1)=0;
        rar3k(1)=sinj3;
        for i=2:k;
            xr3k(i)=qs3(i)*(t-t3(i-1)) + y3(i-1);
            rar3k(i)=rar3(i);
        end;
        xr3k(k+1)=qs3(k+1)*(t-t3(k))+y3(k);
        rar3k(k+1)=rar3(k+1);

```

```

else
    xr3k(1)=0;
    rar3k(1)=1;
    for i=2:k;
        xr3k(i)=qs3(i)*(t-t3(i-1)) + y3(i-1);
        rar3k(i)=rar3(i);
    end;
    xr3k(k+1)=qs3(k+1)*(t-t3(k))+y3(k);
    rar3k(k+1)=rar3(k+1);
end;
ao=min(xr3k);
a1=max(xr3k);
m=frsk/(sk+swi/snom);
a2=m*(t-t1(nt+1)) + y1(nt+1);
a3=interp1(t2,y2,t);
a4=max(qs1)*t;
for i=1:nx;
    if x(i) <= ao;
        satu(i)=sinj3;
        conc(i)=cinj3;
    elseif x(i) >= ao & x(i) <= a1;
        satu(i)=interp1(xr3k,rar3k,x(i));
        conc(i)=cinj3;
    elseif x(i) > a1 & x(i) <= a2;
        satu(i)=interp1(xr2,rar2,x(i));
        conc(i)=cinj2;
    elseif x(i) > a2 & x(i) <= a3;
        satu(i)=s1;
        conc(i)=cinj1;
    elseif x(i) > a3 & x(i) <=a4
        satu(i)=interp1(xr1,rar1,x(i));
        conc(i)=cinj1;
    else
        satu(i)=so;
        conc(i)=co;
    end;
end;
end;
else
clear xr1;clear xr2;

```

```

clear xr3k;clear rar3k;
xr2(1)=0;

for i=2:nt+1;
    xr2(i)=qs2(i)*(t-t1(i-1)) + y1(i-1);
end;
xr1=qs1*t;
y3t=interp1(t3,y3,t);
i=1;
while t >= t3(i);
    k=i;
    i=i+1;
end;
clear xr3k;clear rar3k;

if k < 100
    xr3k(1)=0;
    rar3k(1)=sinj3;
    for i=2:k;
        xr3k(i)=qs3(i)*(t-t3(i-1)) + y3(i-1);
        rar3k(i)=rar3(i);
    end;
    xr3k(k+1)=qs3(k+1)*(t-t3(k))+y3(k);
    rar3k(k+1)=rar3(k+1);
else
    xr3k(1)=0;
    rar3k(1)=1;
    for i=2:k;
        xr3k(i)=qs3(i)*(t-t3(i-1)) + y3(i-1);
        rar3k(i)=rar3(i);
    end;
    xr3k(k+1)=qs3(k+1)*(t-t3(k))+y3(k);
    rar3k(k+1)=rar3(k+1);
end;
foi=interp1(s,fr,so);
fli=interp1(s,fr,s1);
ao=min(xr3k);
a1=max(xr3k);
a2=y3t;

```



```

m=frsk/(sk+swi/snom);
a3=m*(t-t1(nt+1)) + y1(nt+1);
a4=((f1i-foi)/(s1-so))*(t-t2(nt+1))+y2(nt+1);

for i=1:nx;
    if x(i) <= ao
        satu(i)=sinj3;
        conc(i)=cinj3;
    elseif x(i) >ao & x(i) <= a1;
        satu(i)=interp1(xr3k,rar3k,x(i));
        conc(i)=cinj3;
    elseif x(i) > a1 & x(i) <= a2;
        satu(i)=rar3k(k+1);
        conc(i)=cinj3;
    elseif x(i) > a2 & x(i) <= a3;
        satu(i)=interp1(xr2,rar2,x(i));
        conc(i)=cinj2;
    elseif x(i) > a3 & x(i) <= a4
        satu(i)=s1;
        conc(i)=cinj1;
    else
        satu(i)=so;
        conc(i)=co;
    end;
end;
end;

```

else

```

if t <= T0
    clear xr1;
    ao=0;
    xr1=qs1*t;
    a1=max(xr1);
    for i=1:nx;
        if x(i) < ao
            satu(i)=sinj1;
            conc(i)=cinj1;
        elseif x(i) >= ao & x(i) <= a1;

```

```

        satu(i)=interp1(xr1,rar1,x(i));
        conc(i)=co;
    else
        satu(i)=so;
        conc(i)=co;
    end;
end;
elseif T0 < t & t <= T01
    clear xr1;clear xr2k;
    clear rar2k;
    xr1=qs1*t;
    y1t=interp1(t1,y1,t);
    i=1;
    while t >= t1(i);
        k=i;
        i=i+1;
    end;
    raref2t=interp1(t1,rar2,t);
    qs2k=interp1(s,dfj2,raref2t);
    xr2k(1)=0;
    rar2k(1)=sinj2;
    for i=2:k;
        xr2k(i)=qs2(i)*(t-t1(i-1)) + y1(i-1);
        rar2k(i)=rar2(i);
    end;
    xr2k(k+1)=qs2k*(t-t1(k))+y1(k);
    rar2k(k+1)=raref2t;
    ao=0;
    a1=xr2k(k+1);
    a2=y1t;
    a3=max(qs1)*t;
    for i=1:nx;
        if x(i) <= ao
            satu(i)=sinj2;
            conc(i)=cinj2;
        elseif x(i) > ao & x(i) <= a1;
            satu(i)=interp1(xr2k,rar2k,x(i));
            conc(i)=cinj2;
        elseif x(i) > a1 & x(i) <= a2

```

```

        satu(i)=raref2t;
        conc(i)=cinj1;
    elseif x(i) > a2 & x(i) <= a3
        satu(i)=interp1(xr1,rar1,x(i));
        conc(i)=cinj1;
    else
        satu(i)=so;
        conc(i)=co;
    end;
end;

```

```

elseif T01 < t & t <= T02
    clear xr1; clear xr2
    xr1=qs1*t;
    xr2(1)=0;
    for i=2:nt+1;
        xr2(i)=qs2(i)*(t-t1(i)) + y1(i);
    end;
    y2t=interp1(t2,y2,t);
    a0=min(xr2);
    a1=max(xr2);
    a2=y2t;
    a3=max(qs1)*t;
    for i=1:nx;
        if x(i) <= a0
            satu(i)=sinj2;
            conc(i)=cinj2;
        elseif x(i) > a0 & x(i) <= a1
            satu(i)=interp1(xr2,rar2,x(i));
            conc(i)=cinj2;
        elseif x(i) > a1 & x(i) <= a2;
            satu(i)=s1;
            conc(i)=cinj1;
        elseif x(i) > a2 & x(i) <= a3;
            satu(i)=interp1(xr1,rar1,x(i));
            conc(i)=cinj1;
        else
            satu(i)=so;
            conc(i)=co;
        end;
    end;
end;

```

```

        end;
    end;
elseif T02 < t & t <= T1
    clear xr2;
    xr2(1)=0;
    for i=2:nt+1;
        xr2(i)=qs2(i)*(t-t1(i)) + y1(i);
    end;
    frso=interp1(s,fr,so);
    m=(frs1-frso)/(s1-so);
    a0=min(xr2);
    a1=max(xr2);
    a2= m*(t-t2(nt+1)) + y2(nt+1);
    for i=1:nx;
        if x(i) <= a0
            satu(i)=sinj2;
            conc(i)=cinj2;
        elseif x(i) > a0 & x(i) <= a1
            satu(i)=interp1(xr2,raref2,x(i));
            conc(i)=cinj2;
        elseif x(i) > a1 & x(i) <= a2;
            satu(i)=s1;
            conc(i)=cinj1;
        else
            satu(i)=so;
            conc(i)=co;
        end;
    end;
end;
else
    clear xr2;clear qs3k;
    xr2(1)=qs2(1)*t;
    for i=2:nt+1;
        xr2(i)=qs2(i)*(t-t1(i-1)) + y1(i-1);
    end;
    y3t=interp1(t3,y3,t);
    i=1;
    while t >= t3(i);
        k=i;
        i=i+1;
    end;
end;

```

```

end;

if k < 100
    raref3t=interp1(t3,rar3,t);
    qs3k=interp1(s,dfj3,raref3t);
else
    raref3t=rar3(k);
    qs3k=interp1(s,dfj3,raref3t);
end;
clear xr3k;clear rar3k;
if k < 100
    xr3k(1)=qs3(1);
    rar3k(1)=sinj3;
    for i=2:k;
        xr3k(i)=qs3(i)*(t-t3(i-1)) + y3(i-1);
        rar3k(i)=rar3(i);
    end;
    xr3k(k+1)=qs3k*(t-t3(k))+y3(k);
    rar3k(k+1)=raref3t;
else
    xr3k(1)=0;
    rar3k(1)=1;
    for i=2:k;
        xr3k(i)=qs3(i)*(t-t3(i-1)) + y3(i-1);
        rar3k(i)=rar3(i);
    end;
    xr3k(k+1)=qs3k*(t-t3(k))+y3(k);
    rar3k(k+1)=raref3t;
end;
foi=interp1(s,fr,so);
fli=interp1(s,fr,s1);
ao=xr3k(1)*t;
a1=xr3k(k+1);
a2=max(xr2);
a3=((fli-foi)/(s1-so))*(t-t2(nt+1))+y2(nt+1);
for i=1:nx;
    if x(i) <= ao
        satu(i)=sinj3;
        conc(i)=cinj3;
    end;
end;

```

```
elseif x(i) > 0 & x(i) <= a1;
    satu(i)=interp1(xr3k,rar3k,x(i));
    conc(i)=cinj3;
elseif x(i) > a1 & x(i) <= a2;
    satu(i)=interp1(xr2,raref2,x(i));
    conc(i)=cinj2;
elseif x(i) > a2 & x(i) <= a3;
    satu(i)=s1;
    conc(i)=cinj1;
else
    satu(i)=so;
    conc(i)=co;
end;
end;
end;
end;
```

## 16 - Rotina curvalg.m

% Cálculo das curvas de choque do problema

```
clear yo;clear y1;clear y2;  
clear y3;clear y4;clear y5;  
clear to;clear t1;clear t2;  
clear t3;clear t4;clear t5;
```

```
T0=T0/snom;  
T1=T1/snom;  
T2=T2/snom;
```

```
curvay1;  
curvay2;  
curvay3;
```

%curva yo

```
if t2(nt+1)==inf;  
    to=0:.01:10;  
    dfm=interp1(s,dfr,sm);  
    yo=dfm*to;  
else  
    to=0:.01:max(t2);  
    dfm=interp1(s,dfr,sm);  
    yo=dfm*to;  
end;
```

```
if t2(nt+1) < inf  
    t5=t2(nt+1):.01:t2(nt+1)+10;  
    y5=(frs1./(s1 + swi/snom))*(t5-t5(1))+y2(nt+1);  
    t4=t2(1):.1:max(t5);  
    fsk=interp1(s,fr,sk);  
    y4=(fsk./(sk + swi/snom))*(t4-t4(1))+y2(1);  
else  
    t4=t2(1):.1:20;  
    fsk=interp1(s,fr,sk);  
    y4=(fsk./(sk + swi/snom))*(t4-t4(1))+y2(1);  
end;
```

```

T01=Q1;
ds=(sk-sm)./nt;
sd2=sk:-ds:sm;
for i=1:2*nt+1;
    if i < nt+1;
        rar1(i)=sd1(i);
    else
        rar1(i)=sd2(i-nt);
    end;
end;
rar2=sc1;
rar3=sd1;
dfj1=dfr;
dfj2=df1;
dfj3=dfr;
qs1=interp1(s,dfj1,rar1);
qs2=interp1(s,dfj2,rar2);
qs3=interp1(s,dfj3,rar3);

```



## 17 - Rotina curvay1.m

% Algoritmo para o calculo da curva y1(t);

```
y1(1)=0;t1(1)=T0;
ds=(sinj1-sk)./nt;
sd1=sinj1:-ds:sk;
nd1=size(sd1,2);
% Calculo de sel
for j=1:nd1;
    xo=sd1(j);
    if xo >= 1;
        sel(j)=1;
    elseif xo < 1 & xo > sk
        auxi=sstar:(xo-sstar)./nd1:xo;
        nauxi=size(auxi,2);
        fo=interp1(s,fr,xo);
        Mo=fo./(xo + swi/snom);
        r1=1;r2=nauxi;
        k=r2-r1;
        while k > 1;
            r=round((r1+r2)./2);
            fauxi=interp1(s,fl,auxi(r));
            if Mo*(auxi(r)-xo) + fo > fauxi;
                r2=r;
            elseif Mo*(auxi(r)-xo) + fo < fauxi;
                r1=r;
            else
                sel(j)=auxi(r);break;
            end;
            k=r2-r1;
        end;
        sel(j)=(auxi(r1)+auxi(r2))./2;
    else
        sel(j)=sstar;
    end;
end;
nd1=size(sd1,2);
for i=2:nd1;
    fo=interp1(s,fr,sd1(i));
```

```
mo=fo./(sd1(i) + swi/snom);  
m1=interp1(s,dfr,sd1(i));  
t1(i)=(y1(i-1)-mo*t1(i-1))./(m1-mo);  
y1(i)=mo*(t1(i)-t1(i-1))+y1(i-1);  
end;  
  
P1=y1(nd1);Q1=t1(nd1);
```

## 18 - Rotina curvay2.m

% Algoritmo para o cálculo da curva y2

```
if s1 >= sm
    saux=s1;
else
    saux=sm;
end;

clear y2;clear t2;
clear sd2;clear mo;
clear m1;
y2(1)=P1;t2(1)=Q1;
ds=(sk-saux)./nt;
SD2=sk:-ds:saux;
nd2=size(SD2,2);

for i=2:nd2;
    fo=interp1(s,fr,SD2(i));
    f1=interp1(s,fr,s1);
    if SD2(i) > s1;
        mo(i)=(fo-f1)./(SD2(i)-s1) ;
        m1(i)=interp1(s,dfr,SD2(i));
        t2(i)=(y2(i-1)-mo(i)*t2(i-1))./(m1(i)-mo(i));
        y2(i)=mo(i)*(t2(i)-t2(i-1)) + y2(i-1);
    else
        t2(i)=Inf;
        y2(i)=Inf;
    end;
end;
end;
```

## 19 - Rotina curvay3.m

% cálculo da curva y3

f=fl;df=df1;

y3(1)=0;t3(1)=T1;

sd3=se1;

se3=sd1;

nd3=size(sd3,2);

for i=2:nd3-1;

fo=interp1(s,fr,se3(i));

mo(i)=fo./(se3(i) + swi/snom) ;

m1(i)=interp1(s,df,sd3(i));

t3(i)=(y1(i-1)-y3(i-1)-m1(i)\*t1(i-1))+mo(i)\*t3(i-1))./(mo(i)-m1(i));

y3(i)=mo(i)\*(t3(i)-t3(i-1))+y3(i-1);

end;

i=nd3;

mo(i)=fo./(se3(i) + swi/snom) ;

m1(i)=mo(i);

t3(i)=10+t3(i-1);

y3(i)=mo(i)\*(t3(i)-t3(i-1))+y3(i-1);

for i=1:nd3;

t3(i)=abs(t3(i));

y3(i)=abs(y3(i));

end;

## 20 - Rotina inter1.m

% Calcula o estado intermediário s1

```
M=flstar./(sstar + swi/snom);
```

```
r1=1;
```

```
r2=a;
```

```
RETA = M.*(s-sstar)+flstar;
```

```
k=r2-r1;
```

```
while k > 1;
```

```
    r = round((r1+r2)/2);
```

```
    if RETA(r) > fr(r) ;
```

```
        r1=r;
```

```
    else
```

```
        r2=r;
```

```
    end;
```

```
    k=r2-r1;
```

```
end;
```

```
s1= (s(r1)+s(r2))/2;
```

```
frs1=interp1(s,fr,s1);
```

```
dfrs1=interp1(s,dfr,s1);
```

## 21 - Rotina interk.m

% Calcula o estado intermediário sk

```
RETA = M.*(s-sstar)+flstar;
```

```
r1=a;r2=n;
```

```
k=r2-r1;
```

```
while k > 1;
```

```
    r = round((r1+r2)/2);
```

```
    if RETA(r) < fr(r) ;
```

```
        r1=r;
```

```
    else
```

```
        r2=r;
```

```
    end;
```

```
    k=r2-r1;
```

```
end;
```

```
sk=(s(r1)+s(r2))/2;
```

```
frsk=interp1(s,fr,sk);
```

```
dfrsk=interp1(s,dfr,sk);
```

## 22 - Rotina intstar.m

% Calcula o estado intermediário s\*

```
r1=1;r2=n;
k=r2-r1;
snom=1-swi-sor;

while k > 1;
    r=round((r1+r2)/2);
    G = dfl(r)-fl(r)/( s(r) + swi/snom );
    if G > 0 ;
        r1=r;
    else
        r2=r;
    end;
    k= r2-r1;
end;

sstar=(s(r1)+s(r2))/2;
flstar=interp1(s,fl,sstar);
dflstar=interp1(s,dfl,sstar);
a=r1;
b=r2;
```

### 23 - Rotina intersm.m

% Calcula o estado intermediário sm

```
r1=1;r2=n;
```

```
k=r2-r1;
```

```
while k > 1;
```

```
    r=round((r1+r2)/2);
```

```
    G = dfr(r) - fr(r)/s(r);
```

```
    if G > 0 ;
```

```
        r1=r;
```

```
    else
```

```
        r2=r;
```

```
    end;
```

```
    k= r2-r1;
```

```
end;
```

```
sm=(s(r1)+s(r2))/2;
```

```
frsm=interp1(s,fr,sm);
```

```
dfrsm=interp1(s,dfr,sm);
```



## 24 - Rotina prodbanc.m

```
% Calcula o perfil de saturação s(x,t) e concentração c(x,t)
% e a produção
% Devemos entrar com os seguintes valor tmax
```

```
tmax=tdmax/(1-swi-sor);
```

```
if tmax == 0
```

```
    producao=0;
```

```
else
```

```
    clear A; clear B;clear X;
```

```
    T01=t1(nt+1);T02=max(t2);
```

```
    ntmax=10*10;
```

```
    dt=tmax/ntmax;
```

```
    tempo=0:dt:tmax;size(tempo,2);
```

```
    for j=2:size(tempo,2);
```

```
        t=tempo(j);
```

```
        if T1 < T02
```

```
            if t <= T0 ;
```

```
                clear xr1;
```

```
                ao=0;
```

```
                xr1=qs1*t;
```

```
                al=max(xr1);
```

```
                for i=1:nx;
```

```
                    if x(i) >= ao & x(i) <= al;
```

```
                        satu(i)=interp1(xr1,rar1,x(i));
```

```
                        conc(i)=co;
```

```
                    else
```

```
                        satu(i)=so;
```

```
                        conc(i)=co;
```

```
                    end;
```

```
                end;
```

```
                A(j,:)=satu;B(j,:)=conc;
```

```
            elseif T0 < t & t <= T01
```

```
                clear xr1;clear xr2k;clear rar2k;
```

```
                xr1=qs1*t;
```

```
                y1t=interp1(t1,y1,t);
```

```
                i=1;
```

```
                while t >= t1(i);
```

```

        k=i;
        i=i+1;
    end;
    raref2t=interp1(t1,rar2,t);
    qs2k=interp1(s,dfj2,raref2t);
    xr2k(1)=0;
    rar2k(1)=sinj2;
    for i=2:k;
        xr2k(i)=qs2(i)*(t-t1(i-1)) + y1(i-1);
        rar2k(i)=rar2(i);
    end;
    xr2k(k+1)=qs2k*(t-t1(k))+y1(k);
    rar2k(k+1)=raref2t;
    ao=0;
    a1=xr2k(k+1);
    a2=y1t;
    a3=max(qs1)*t;
    for i=1:nx;
        if x(i) <= ao
            satu(i)=sinj2;
            conc(i)=cinj2;
        elseif x(i) > ao & x(i) <= a1;
            satu(i)=interp1(xr2k,rar2k,x(i));
            conc(i)=cinj2;
        elseif x(i) > a1 & x(i) <= a2
            satu(i)=raref2t;
            conc(i)=cinj1;
        elseif x(i) > a2 & x(i) <= a3
            satu(i)=interp1(xr1,rar1,x(i));
            conc(i)=cinj1;
        else
            satu(i)=so;
            conc(i)=co;
        end;
    end;
    A(j,:)=satu;B(j,:)=conc;
elseif T01 < t & t <= T1
    clear xr1; clear xr2
    xr1=qs1*t;

```

```

xr2(1)=0;
for i=2:nt+1;
    xr2(i)=qs2(i)*(t-t1(i)) + y1(i);
end;
y2t=interp1(t2,y2,t);
a0=min(xr2);
a1=max(xr2);
a2=y2t;
a3=max(qs1)*t;
for i=1:nx;
    if x(i) <= a0
        satu(i)=sinj2;
        conc(i)=cinj2;
    elseif x(i) > a0 & x(i) <= a1
        satu(i)=interp1(xr2,rar2,x(i));
        conc(i)=cinj2;
    elseif x(i) > a1 & x(i) <= a2
        satu(i)=s1;
        conc(i)=cinj1;
    elseif x(i) > a2 & x(i) <= a3;
        satu(i)=interp1(xr1,rar1,x(i));
        conc(i)=cinj1;
    else
        satu(i)=so;
        conc(i)=co;
    end;
end;
A(j,:)=satu;B(j,:)=conc;
elseif T1 < t & t <= T02
clear xr1;clear xr2;
clear xr3k;clear rar3k;
xr2(1)=qs1(1)*t;
for i=2:nt+1;
    xr2(i)=qs2(i)*(t-t1(i-1)) + y1(i-1);
end;

xr1=qs1*t;
y3t=interp1(t3,y3,t);
se3t=interp1(t3,se3,t);

```

```

sd3t=interp1(t3,sd3,t);
i=1;
while t >= t3(i);
    k=i;
    i=i+1;
end;
if k < 100
    raref3t=interp1(t3,rar3,t);
    qs3k=interp1(s,dfr,raref3t);
else
    raref3t=rar3(k) ;
    qs3k=interp1(s,dfr,raref3t);
end;
clear xr3k;clear rar3k;
if k < 100
    xr3k(1)=0;
    rar3k(1)=sinj3;
    for i=2:k;
        xr3k(i)=qs3(i)*(t-t3(i)) + y3(i);
        rar3k(i)=rar3(i);
    end;
    xr3k(k+1)=qs3k*(t-t3(k))+y3(k);
    rar3k(k+1)=raref3t;
else
    xr3k(1)=0;
    rar3k(1)=1;
    for i=2:k;
        xr3k(i)=qs3(i)*(t-t3(i-1)) + y3(i-1);
        rar3k(i)=rar3(i);
    end;
    xr3k(k+1)=qs3k*(t-t3(k))+y3(k);
    rar3k(k+1)=raref3t;
end;
y2t=interp1(t2,y2,t);
ao=xr3k(1);
a1=xr3k(k+1);
a2=y3t;
a3=max(xr2);
a4=y2t;

```

```

a5=max(qs1)*t;
for i=1:nx;
    if x(i) <= ao;
        satu(i)=sinj3;
        conc(i)=cinj3;
    elseif x(i) >= ao & x(i) <= a1;
        satu(i)=interp1(xr3k,rar3k,x(i));
        conc(i)=cinj3;
    elseif x(i) > a1 & x(i) <= a2;
        satu(i)=se3t;
        conc(i)=cinj3;
    elseif x(i) > a2 & x(i) <= a3;
        satu(i)=interp1(xr2,rar2,x(i));
        conc(i)=cinj2;
    elseif x(i) > a3 & x(i) <=a4
        satu(i)=s1;
        conc(i)=cinj1;
    elseif x(i) > a4 & x(i) <= a5;
        satu(i)=interp1(xr1,rar1,x(i));
        conc(i)=cinj1;
    else
        satu(i)=so;
        conc(i)=co;
    end;
end;
A(j,:)=satu;B(j,:)=conc;
else
clear xr2;clear qs3k;
xr2(1)=qs2(1)*t;
for i=2:nt+1;
    xr2(i)=qs2(i)*(t-t1(i-1)) + y1(i-1);
end;
y3t=interp1(t3,y3,t);
se3t=interp1(t3,se3,t);
sd3t=interp1(t3,sd3,t);
i=1;
while t >= t3(i);
    k=i;
    i=i+1;

```

```

end;
if k < 100
    raref3t=interp1(t3,rar3,t);
    qs3k=interp1(s,dfj3,raref3t);
else
    raref3t=rar3(k);
    qs3k=interp1(s,dfj3,raref3t);
end;
clear xr3k;clear rar3k;
if k < 100
    xr3k(1)=qs3(1)*t;
    rar3k(1)=sinj3;
    for i=2:k;
        xr3k(i)=qs3(i)*(t-t3(i-1)) + y3(i-1);
        rar3k(i)=rar3(i);
    end;
    xr3k(k+1)=qs3k*(t-t3(k))+y3(k);
    rar3k(k+1)=raref3t;
else
    xr3k(1)=0;
    rar3k(1)=1;
    for i=2:k;
        xr3k(i)=qs3(i)*(t-t3(i-1)) + y3(i-1);
        rar3k(i)=rar3(i);
    end;
    xr3k(k+1)=qs3k*(t-t3(k))+y3(k);
    rar3k(k+1)=raref3t;
end;
foi=interp1(s,fr,so);
fli=interp1(s,fr,s1);
ao=xr3k(1);
a1=xr3k(k+1);
a2=y3t;
a3=max(xr2);
a4=((fli-foi)/(s1-so))*(t-t2(nt+1))+y2(nt+1);
for i=1:nx;
    if x(i) <= ao
        satu(i)=sinj3;
        conc(i)=cinj3;
    end;
end;

```

```

elseif x(i) > ao & x(i) <= a1;
    satu(i)=interp1(xr3k,rar3k,x(i));
    conc(i)=cinj3;
elseif x(i) > a1 & x(i) <= a2;
    satu(i)=se3t;
    conc(i)=cinj3;
elseif x(i) > a2 & x(i) <= a3;
    satu(i)=interp1(xr2,rar2,x(i));
    conc(i)=cinj2;
elseif x(i) > a3 & x(i) <= a4;
    satu(i)=s1;
    conc(i)=cinj1;
else
    satu(i)=so;
    conc(i)=co;
end;
end;
A(j,:)=satu;B(j,:)=conc;
end;
else
if t <= T0
clear xrl;
ao=0;
xrl=qs1*t;
a1=max(xrl);
for i=1:nx;
    if x(i) < ao
        satu(i)=sinj1;
        conc(i)=cinj1;
    elseif x(i) >= ao & x(i) <= a1;
        satu(i)=interp1(xrl,rar1,x(i));
        conc(i)=co;
    else
        satu(i)=so;
        conc(i)=co;
    end;
end;
end;
A(j,:)=satu;B(j,:)=conc;
elseif T0 < t & t <= T01

```

```

clear xr1;clear xr2k;clear rar2k;
xr1=qs1*t;
y1t=interp1(t1,y1,t);
i=1;
while t >= t1(i);
    k=i;
    i=i+1;
end;
raref2t=interp1(t1,rar2,t);
qs2k=interp1(s,dfj2,raref2t);
xr2k(1)=0;
rar2k(1)=sinj2;
for i=2:k;
    xr2k(i)=qs2(i)*(t-t1(i-1)) + y1(i-1);
    rar2k(i)=rar2(i);
end;
xr2k(k+1)=qs2k*(t-t1(k))+y1(k);
rar2k(k+1)=raref2t;
ao=0;
a1=xr2k(k+1);
a2=y1t;
a3=max(qs1)*t;

for i=1:nx;
    if x(i) <= ao
        satu(i)=sinj2;
        conc(i)=cinj2;
    elseif x(i) > ao & x(i) <= a1;
        satu(i)=interp1(xr2k,rar2k,x(i));
        conc(i)=cinj2;
    elseif x(i) > a1 & x(i) <= a2
        satu(i)=raref2t;
        conc(i)=cinj1;
    elseif x(i) > a2 & x(i) <= a3
        satu(i)=interp1(xr1,rar1,x(i));
        conc(i)=cinj1;
    else
        satu(i)=so;
        conc(i)=co;
    end
end

```



```

        end;
    end;
    A(j,:)=satu;B(j,:)=conc;
elseif T01 < t & t <= T02
    clear xr1; clear xr2
    xr1=qs1*t;
    xr2(1)=0;
    for i=2:nt+1;
        xr2(i)=qs2(i)*(t-t1(i)) + y1(i);
    end;
    y2t=interp1(t2,y2,t);
    a0=min(xr2);
    a1=max(xr2);
    a2=y2t;
    a3=max(qs1)*t;

    for i=1:nx;
        if x(i) <= a0
            satu(i)=sinj2;
            conc(i)=cinj2;
        elseif x(i) > a0 & x(i) <= a1
            satu(i)=interp1(xr2,rar2,x(i));
            conc(i)=cinj2;
        elseif x(i) > a1 & x(i) <= a2;
            satu(i)=s1;
            conc(i)=cinj1;
        elseif x(i) > a2 & x(i) <= a3;
            satu(i)=interp1(xr1,rar1,x(i));
            conc(i)=cinj1;
        else
            satu(i)=so;
            conc(i)=co;
        end;
    end;
    A(j,:)=satu;B(j,:)=conc;
elseif T02 < t & t <= T1
    clear xr2;
    xr2(1)=0;
    for i=2:nt+1;

```

```

        xr2(i)=qs2(i)*(t-t1(i)) + y1(i);
    end;
    frso=interp1(s,fr,so);
    m=(frs1-frso)/(s1-so);
    a0=min(xr2);
    a1=max(xr2);
    a2= m*(t-t2(nt+1)) + y2(nt+1);

    for i=1:nx;
        if x(i) <= a0
            satu(i)=sinj2;
            conc(i)=cinj2;
        elseif x(i) > a0 & x(i) <= a1
            satu(i)=interp1(xr2,rar2,x(i));
            conc(i)=cinj2;
        elseif x(i) > a1 & x(i) <= a2;
            satu(i)=s1;
            conc(i)=cinj1;
        else
            satu(i)=so;
            conc(i)=co;
        end;
    end;
    A(j,:)=satu;B(j,:)=conc;
else
    clear xr2;clear qs3k;
    xr2(1)=qs2(1)*t;
    for i=2:nt+1;
        xr2(i)=qs2(i)*(t-t1(i-1)) + y1(i-1);
    end;
    y3t=interp1(t3,y3,t);
    i=1;
    while t >= t3(i);
        k=i;
        i=i+1;
    end;
    if k < 100
        raref3t=interp1(t3,rar3,t);
        qs3k=interp1(s,dfj3,raref3t);

```

```

else
    raref3t=rar3(k);
    qs3k=interp1(s,dfj3,raref3t);
end;
clear xr3k;clear rar3k;
if k < 100
    xr3k(1)=qs3(1);
    rar3k(1)=sinj3;
    for i=2:k;
        xr3k(i)=qs3(i)*(t-t3(i-1)) + y3(i-1);
        rar3k(i)=rar3(i);
    end;
    xr3k(k+1)=qs3k*(t-t3(k))+y3(k);
    rar3k(k+1)=raref3t;
else
    xr3k(1)=0;
    rar3k(1)=1;
    for i=2:k;
        xr3k(i)=qs3(i)*(t-t3(i-1)) + y3(i-1);
        rar3k(i)=rar3(i);
    end;
    xr3k(k+1)=qs3k*(t-t3(k))+y3(k);
    rar3k(k+1)=raref3t;
end;
foi=interp1(s,fr,so);
fli=interp1(s,fr,s1);
ao=xr3k(1)*t;
a1=xr3k(k+1);
a2=max(xr2);
a3=((fli-foi)/(s1-so))*(t-t2(nt+1))+y2(nt+1);
for i=1:nx;
    if x(i) <= ao
        satu(i)=sinj3;
        conc(i)=cinj3;
    elseif x(i) > 0 & x(i) <= a1;
        satu(i)=interp1(xr3k,rar3k,x(i));
        conc(i)=cinj3;
    elseif x(i) > a1 & x(i) <= a2;
        satu(i)=interp1(xr2,rar2,x(i));

```

```

        conc(i)=cinj2;
    elseif x(i) > a2 & x(i) <= a3;
        satu(i)=s1;
        conc(i)=cinj1;
    else
        satu(i)=so;
        conc(i)=co;
    end;
end;
A(j,:)=satu;B(j,:)=conc;
end
end
end
nx=size(A,2);
F0=1-fract(A(:,nx),B(:,nx));
producao=trapz(tempo,F0);

% Animação gráfica mostra a evolução dos perfis da solução com o tempo
ANI=input('Para uma animação gráfica aperte 1 , caso contrário aperte 0 ' );
ANI=0;
if ANI == 1
    close;ntempo=size(tempo,2);
    M=moviein(ntempo);
    for j=1:ntempo;plot(x,A(j,:),x,B(j,:));
        M(:,j)=getframe;
    end;
    pause;
    mesh(A);
else
end;
end;

```

## 25 - Rotina curvas.m

```
% T0 : tempo onde termina a injeção do primeiro banco e começa a injeção  
% do segundo banco.  
% T1 : tempo onde termina a injeção do segundo banco e começa a injeção  
% do terceiro banco.
```

```
T0=T0/snom;
```

```
T1=T1/snom;
```

```
%cálculo da curva y11
```

```
clear y11;clear t11;
```

```
nr=size(leq11,2);
```

```
y11(1)=0;t11(1)=T0;
```

```
for j=2:nr;
```

```
    m1=interp1(s,dfr,leq11(j));
```

```
    f0=interp1(s,fr,leq11(j));
```

```
    m2=f0./(leq11(j)+h1r);
```

```
    d=y11(j-1)-m2*t11(j-1);
```

```
    t11(j)=d./(m1-m2);
```

```
    y11(j)=m1*t11(j);
```

```
end;
```

```
%cálculo da curva y12
```

```
if s1 <= sm
```

```
    clear y12;clear t12;
```

```
    y12(1)=y11(nr);t12(1)=t11(nr);
```

```
    for j=2:nr;
```

```
        m1=interp1(s,dfr,leq12(j));
```

```
        f0=interp1(s,fr,leq12(j));
```

```
        m2=(frs1-f0)/(s1-leq12(j));
```

```
        d=y12(j-1)-m2*t12(j-1);
```

```
        t12(j)=d./(m1-m2);
```

```
        y12(j)=m1*t12(j);
```

```
    end;
```

```
else
```

```
    y12(1)=y11(nr);t12(1)=t11(nr);
```

```

aux=sk:(s1-sk)./nt : s1;
for j=2:nr-1;
    m1=interp1(s,dfr,aux(j));
    f0=interp1(s,fr,aux(j))+(10)^(-10);
    m2=(frs1-f0)/(s1-aux(j));
    d=y12(j-1)-m2*t12(j-1);
    t12(j)=d./(m1-m2);
    y12(j)=m1*t12(j);
end;
t12(nr)=inf;
y12(nr)=inf;
end;
% a curva y13;
if s1 <= sm
    clear y13;clear t13;
    frso=interp1(s,fr,so);
    t13=t12(nr):.1:10+t12(nr);
    m=(frs1-frso)/(s1-so);
    y13=m*(t13-t12(nr))+y12(nr);

% a curva y14;
clear y14;clear t14;
t14=t11(nr):.01:t13(nr);
m=df1star;
y14=m*(t14-t11(nr))+y11(nr);
% a curva y15;
clear y15;clear t15;
t15=0:.01:t12(nr);
y15=dfrsm*t15;
else

% a curva y14;
clear y14;clear t14;
t14=t11(nr):.01:t12(nr-1);
h1=h(k1,k2,cinj2)+ swi./snom;
m=df1star;
y14=m*(t14-t11(nr))+y11(nr);

% a curva y15;

```

```
clear y15;clear t15;  
t15=0:.01:t12(nr-1);  
y15=dfism*t15;
```

```
end;
```

## 26 - Rotina prodads.m

```
% Calcula a produção para o caso da injeção de bancos com  
% adsorção
```

```
clear satu;  
clear conc;  
clear A;  
clear B;
```

```
T01=t11(nt+1);  
T11=t12(nt+1);
```

```
tmax=tdmax/snom;  
ntmax=50;  
dt=tmax/ntmax;
```

```
if tmax == 0;  
    producao=0;
```

```
else
```

```
    tempo=0:dt:tmax;
```

```
    for k=1:ntmax+1;
```

```
        t=tempo(k);
```

```
        twobanc;
```

```
        A(k,:)=satu;
```

```
        B(k,:)=conc;
```

```
    end;
```

```
    Fadsor=1-fract(A(:,nx),B(:,nx));
```

```
    producao=trapz(tempo,Fadsor);
```

```
end;
```



## 27 - Rotina twobanc.m

% Calcula a solução  $s(x,t)$ ,  $c(x,t)$  para um valor de  $t$   
% Devemos então entrar com o valor de  $t$

```
clear satu;
clear conc;

T01=t11(nt+1);
T11=t12(nt+1);

if t <=T0;
    clear leq1;
    clear xleq1;
    ds=(sinj1-sm)./nt;
    leq1=sinj1:-ds:sm;
    qs=interp1(s,dfr,leq1);
    xleq1=qs*t;
    a1=max(xleq1);
    ao=xleq1(1);
    for i=1:nx;
        if x(i) >=ao & x(i) <= a1;
            satu(i)=interp1(xleq1,leq1,x(i));
            conc(i)=cinj1;
        else
            satu(i)=so;
            conc(i)=co;
        end;
    end;
    A(k,:)=satu;B(k,:)=conc;
elseif t > T0 & t <= T01;
    clear leq2t;
    clear xleq2t;
    j=1;
    while t11(j) < t ;
        nrt=j ;
        j=j+1;
    end;
    for j=1:nrt;
        leq2t(j)=leq2(j);
```

```

end;
leq2t(nrt+1)=interp1(t11,leq2,t);
for j=2:nrt;
    m(j)=interp1(s,dfl,leq2t(j));
    xleq2t(j)=m(j)*(t-t11(j))+y11(j);
end;
xleq2t(nrt+1)=interp1(t11,y11,t);
xleq2t(1)=0;
ds=(sinj1-sm)./nt;
leq=sinj1:-ds:sm;
D=interp1(s,dfr,leq);
xleq=D*t;
a1=xleq2t(1);
a2=xleq2t(nrt+1);
a3=max(xleq);
for i=1:nx;
    if x(i) < a1;
        satu(i)=sinj2;
        conc(i)=cinj2;
    elseif x(i) >= a1 & x(i) < a2;
        satu(i)=interp1(xleq2t,leq2t,x(i));
        conc(i)=cinj2;
    elseif x(i) >= a2 & x(i) < a3;
        satu(i)=interp1(xleq,leq,x(i));
        conc(i)=cinj1;
    else
        satu(i)=so;
        conc(i)=co;
    end;
end;
A(k,:)=satu;B(k,:)=conc;
else
clear xleq2;
for j=1:nt+1;
    m(j)=interp1(s,dfl,leq2(j));
    xleq2(j)=m(j)*(t-t11(j))+y11(j);
end;
ds=(sinj1-sm)./nt;
leq=sinj1:-ds:sm;

```

```

D=interp1(s,dfr,leq);
xleq=D*t;
a1=xleq2(1);
a2=max(xleq2);
a3=interp1(t12,y12,t);
a4=max(xleq);
for i=1:nx;
    if x(i) < a1;
        satu(i)=sinj2;
        conc(i)=cinj2;
    elseif x(i) >= a1 & x(i) < a2;
        satu(i)=interp1(xleq2,leq2,x(i));
        conc(i)=cinj2;
    elseif x(i) >= a2 & x(i) < a3;
        satu(i)=s1;
        conc(i)=cinj1;
    elseif x(i) >= a3 & x(i) < a4;
        satu(i)=interp1(xleq,leq,x(i));
        conc(i)=co;
    else
        satu(i)=so;
        conc(i)=co;
    end;
end;
A(k,:)=satu;B(k,:)=conc;
end;

```

## 28 - Rotina joha.m

% Dá a solução  $s(x,t)$  e  $c(x,t)$  descrita por Johansen para o  
% caso  $c_l > c_r$

```
clear satu;clear conc;
intstar;
interl;
interk;

if SL < sstar
    %cálculo de  $u_1=(S_1,CR)$ ;
    fsl=interp1(s,fl,SL);
    M=fsl/(SL+hlr);
    r1=1;
    r2=nstar;
    RETA = M.*(s-SL)+fsl;
    k=r2-r1;
    while k > 1;
        r = round((r1+r2)/2);
        if RETA(r) > fr(r) ;
            r1=r;
        else
            r2=r;
        end;
        k=r2-r1;
    end;
    S1= (s(r1)+s(r2))/2;
    xs=S1;F=fr;
    rotsku;skul=skxu;
    if SR < skul;
        se=S1;
        sd=SR;
        f=fr;
        df=dfr;
        buck;
        v1=fsl/(S1+hlr);
        v2=vi;
        v3=vf;
        for i=1:nx;
```

```

    if x(i) < v1*t
        satu(i)=SL;
        conc(i)=CL;
    elseif x(i) >= v1*t & x(i) < v2*t
        satu(i)=S1;
        conc(i)=CR;
    elseif x(i) >= v2*t & x(i) < v3*t
        satu(i)=satura(i);
        conc(i)=CR;
    else
        satu(i)=SR;
        conc(i)=CR;
    end;
end;
elseif SR > skul
    %cálculo de u2=(S2,CL);
    fsr=interp1(s,fr,SR);
    M=fsr/(SR+h1r);
    r1=nstar;
    r2=n;   RETA = M.*(s-SR)+fsr;
    k=r2-r1;
    while k > 1;
        r = round((r1+r2)/2);
        if RETA(r) > fl(r) ;
            r2=r;
        else
            r1=r;
        end;
        k=r2-r1;
    end;
    S2= (s(r1)+s(r2))/2;
    se=SL;
    sd=S2;
    f=fl;
    df=df1;
    buck;
    v3=M;
    v1=vi;
    v2=vf;

```

```

for i=1:nx;
    if x(i) < v1*t
        satu(i)=SL;
        conc(i)=CL;
    elseif x(i) >= v1*t & x(i) < v2*t
        satu(i)=satura(i);
        conc(i)=CL;
    elseif x(i) >= v2*t & x(i) < v3*t
        satu(i)=S2;
        conc(i)=CL;
    else
        satu(i)=SR;
        conc(i)=CR;
    end;
end;
else
    fsr=interp1(s,fr,SR);
    v1=fsr/(SR+h1r);
    for i=1:nx;
        if x(i) < v1*t
            satu(i)=SL;
            conc(i)=CL;
        else
            satu(i)=SR;
            conc(i)=CR;
        end;
    end;
end;
else
    if SR >= sk
        fsr=interp1(s,fr,SR);
        M=fsr/(SR+h1r);
        r1=nstar;
        r2=n;
        RETA = M.*(s-SR)+fsr;
        k=r2-r1;
        while k > 1;
            r = round((r1+r2)/2);
            if RETA(r) > fl(r) ;

```

```

        r2=r;
    else
        r1=r;
    end;
    k=r2-r1;
end;
S2= (s(r1)+s(r2))/2;
se=SL;
sd=S2;
f=fl;
df=df1;
buck;
v3=M;
v1=vi;
v2=vf;
for i=1:nx;
    if x(i) < v1*t
        satu(i)=SL;
        conc(i)=CL;
    elseif x(i) >= v1*t & x(i) < v2*t
        satu(i)=satura(i);
        conc(i)=CL;
    elseif x(i) >= v2*t & x(i) < v3*t
        satu(i)=S2;
        conc(i)=CL;
    else
        satu(i)=SR;
        conc(i)=CR;
    end;
end;
else
se=SL;
sd=sstar;
f=fl;
df=df1;
buck;
v1=vi;
v2=vf;
satura1=satura;

```

```

se=s1;
sd=SR;
f=fr;
df=dfr;
buck;
satura2=satura;
v3=vi;
for i=1:nx;
    if x(i) < v1*t ;
        satu(i)=SL;
        conc(i)=CL;
    elseif x(i) >= v1*t & x(i) < v2*t;
        satu(i)=satura1(i);
        conc(i)=CL;
    elseif x(i) >= v2*t & x(i) < v3*t ;
        satu(i)=satura2(i);
        conc(i)=CR;
    else
        satu(i)=SR;
        conc(i)=CR;
    end;
end;
end;
end;
end;

```



## 29 - Rotina produjoh.m

% Calcula a produção para o caso da injeção de água com polímeros (Johansen)  
% Entre com o valor para a variável tdmx  
% Deve ser executado depois de executada a rotina dados.m

```
clear A;clear B;
if tdmx < 1
    jt=10;
else
    jt=fix(tdmx);
end;
snom=1-swi-sor;
tmax=tdmx/snom;
ntmax=jt*100;
dtm=tmax/ntmax;

if tmax == 0
    producao=0;
else
    tempo=0:dtm:tmax;
    ntemp=size(tempo,2);
    for j1=1:ntemp;
        t=tempo(j1);
        joha;
        A(j1,:)=satu;B(j1,:)=conc;
    end;
    Fjoha=1-fract(A(:,nx),B(:,nx));
    producao=trapz(tempo,Fjoha);
end;
```

### 30 - Rotina rotsku.m

```
% Calcula sk(u) dado na teoria.
snom=1-swi-sor;
cr=CR;cl=CL;
ar= .2*cr./(1+100*cr);
al= .2*cl./(1+100*cl);
hlr=(ar-al)/(cr-cl);
hlr=(swi+hlr)/snom ;
fsx=interp1(s,F,xs);
Mx=fsx/(xs+hlr);
N=1/(1+hlr);
if Mx < N
    skxu=inf;
    % sk=inf significa que a reta com inclinação lampc(x,y)
    % corta o gráfico de f em um só ponto ,no caso u=(x,y)
else
    RETA = Mx.*(s+hlr);
    if xs < sstar
        r1=nstar;r2=n;
        k=r2-r1;
        while k > 1;
            r = round((r1+r2)/2);
            fsr=interp1(s,F,s(r));
            G = fsr - RETA(r);
            if G > 0 ;
                r1=r;
            else
                r2=r;
            end;
            k=r2-r1;
        end;
        skxu=(s(r1)+s(r2))/2;
        fskxu=interp1(s,F,skxu);
    else
        r2=nstar;r1=1;
    end;
    k=r2-r1;
    while k > 1;
        r = round((r1+r2)/2);
```

```
fsr=interp1(s,F,s(r));
G = fsr - RETA(r);
if G > 0 ;
    r1=r;
else
    r2=r;
end;
k=r2-r1;
end;
skxu=(s(r1)+s(r2))/2;
fkxu=interp1(s,F,skxu);
end;
```

### 31 – Definição das funções utilizadas.

#### Function fract.m

```
% A função de fluxo fracionário f(s,c)
function f = fract(s,c);
f = (s.^2)./(s.^2 + (0.5 + 100*c).*((1-s).^2));
end;
```

#### Function derfract.m

```
% Derivada parcial de f(s,c) com relação a s
function z=dfract(x,y);
z = (2.*x.*(1-x).*(0.5+100*y))./( ( x.^2 + (0.5+100*y).*(1-x).^2 ).^2) ;
end;
```

#### Function dfc1.m

```
% Derivada parcial de f(s,c) com relação a c
function f = dfc(s,c);
f = -100*((s.^2)*(1-s)^2 )./ ( (s.^2 + (0.5 +100*c).*((1-s).^2)).^2);
end;
```

#### Function lampc1.m

```
% Velocidade do c-choque
function yp = lampc(s,c,swi,sor);
f1= (s.*s) ./ (s.*s + (0.5 + 100*c).*((1-s).*(1-s)) );
snom=1-swi-sor;
yp = f1./(s+swi/snom);
end;
```

#### Function h.m

```
% h = h(c) : derivada de a(c)
function yp= h(k1,k2,x);
yp=(k1)./(1+k2.*x).^2 ;
end;
```