Escola Politécnica da Universidade de São Paulo



# MAP3121 - Métodos Numéricos e Aplicações

Turma 4 - Prof. Marcone

## Projeto: Obtenção de Distribuição de Temperatura

### Integrante:

Lucas Canossa Cipolla, 10769542

# 1. Introdução

Nesse projeto, utilizamos a teoria dos problemas inversos para resolução do mesmo. Nesta teoria, observamos o efeito causado para então descobrirmos a causa, uma maneira invertida para resolver problemas. Nossa proposta é uma equação diferencial parcial da física, onde veremos que é possível calcular todo seu efeito apenas sabendo as condições de contorno.

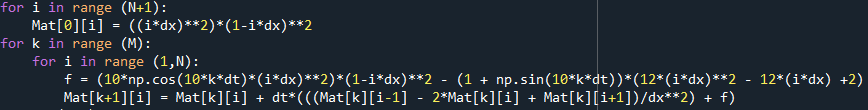
# 2. Descrição do problema

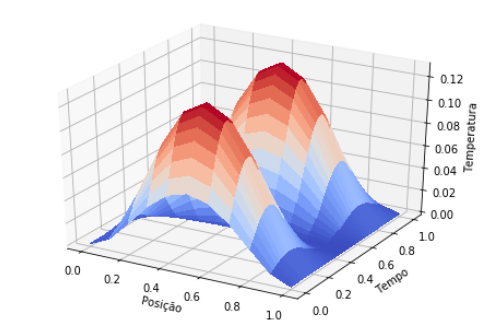
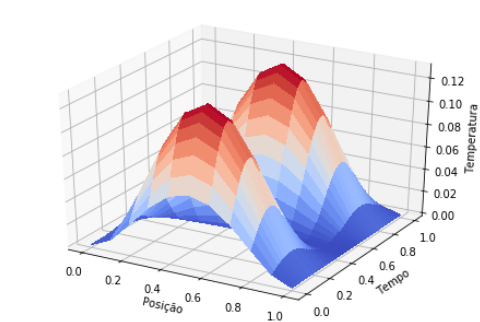
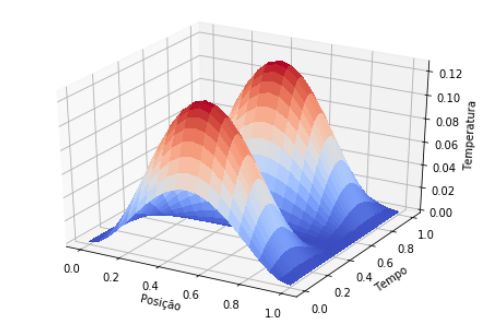
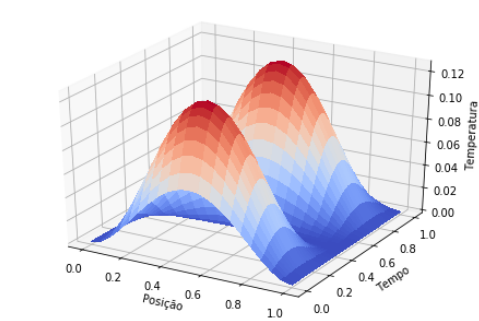
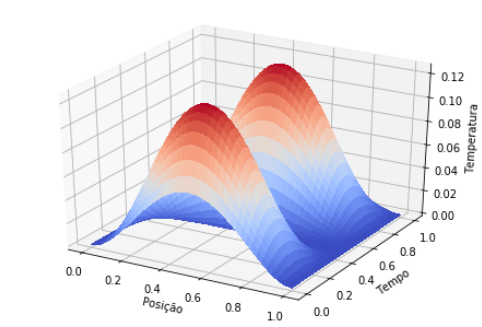
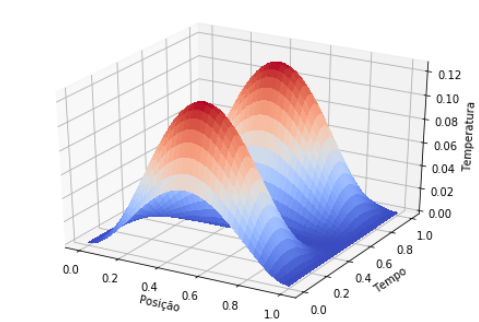
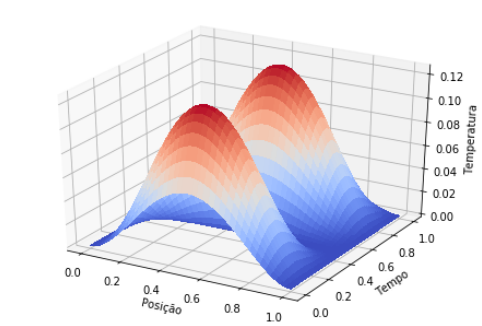
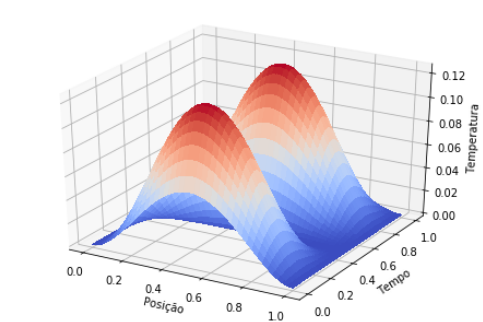
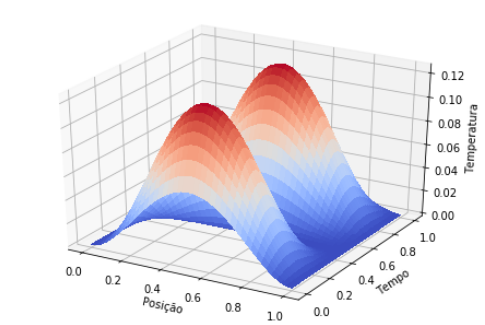
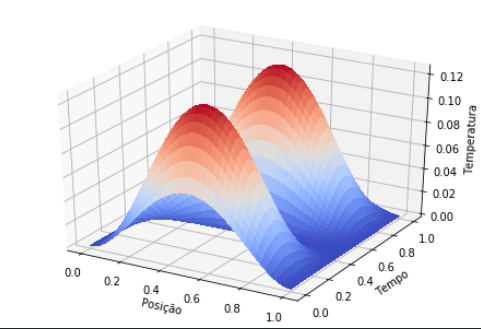
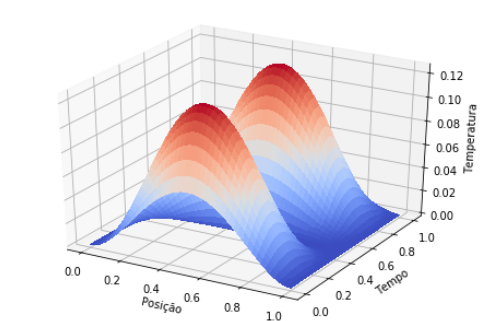
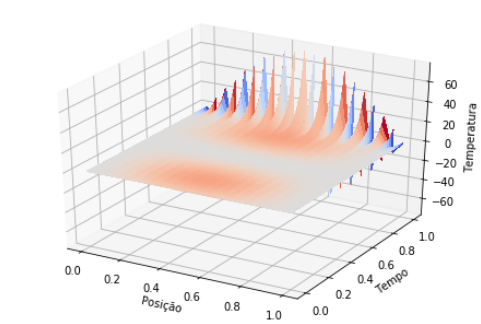
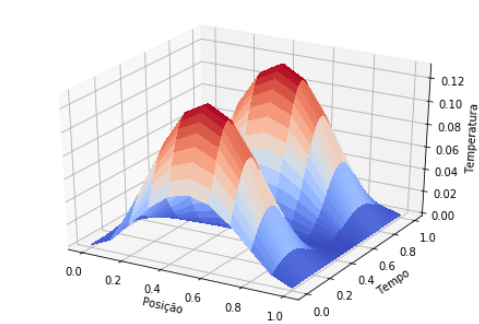
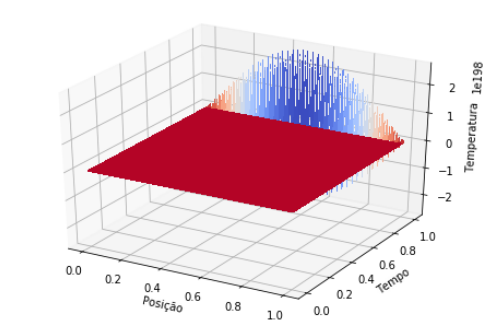
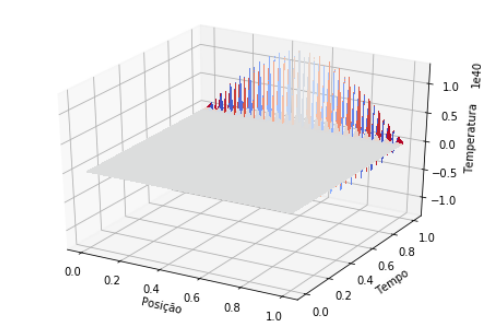
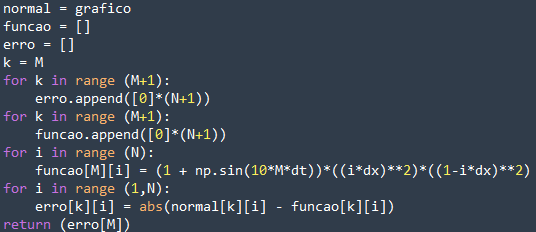
Neste relatório, será utilizada a equação do calor, ou da distribuição de temperatura. É premissa da física que esta é descrita por quatro condições:  
  
ut(t, x) = uxx(t, x) + f(t, x) em [0, T] × [0, 1]  
u(0, x) = u0(x) em [0, 1]  
u(t, 0) = g1(t) em [0, T]  
u(t, 1) = g2(t) em [0, T].  
  
 Além disso, conseguimos calcular a evolução em pontos inferiores, pela fórmula a seguir:  

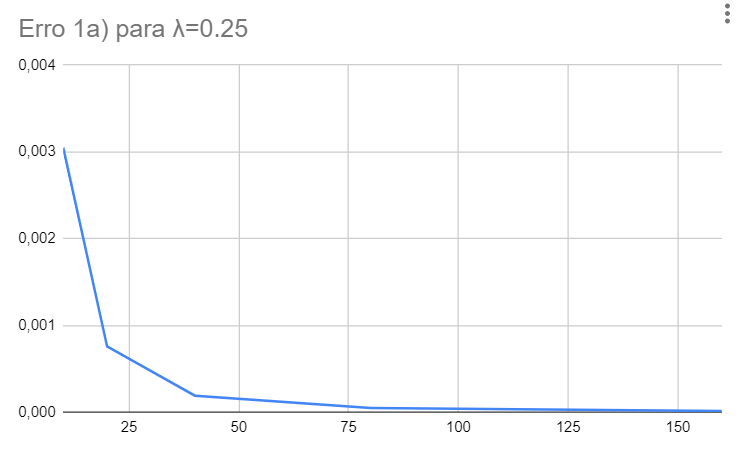
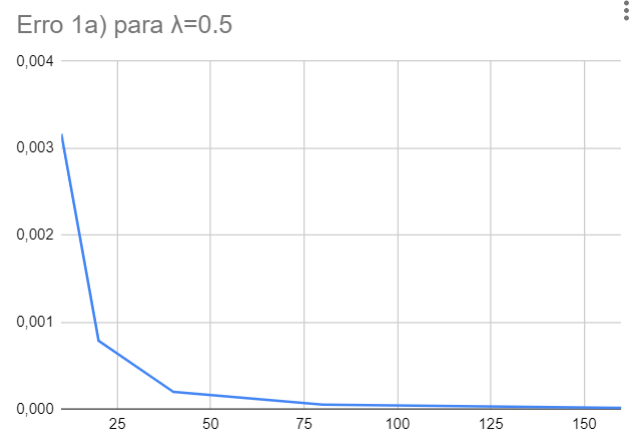

Ainda assim, haverá certo erro perante a ideia inicial, que será definido como o módulo do máximo erro dentro todos os pontos do nosso experimento contra a ideia ().

Além disso, nossa demonstração evidencia um lambida que segue a proposta a seguir, que se desrespeitada falha com o experimento (λ = ∆t/∆x^2 ≤ ½) .

# 3. Tarefa 1

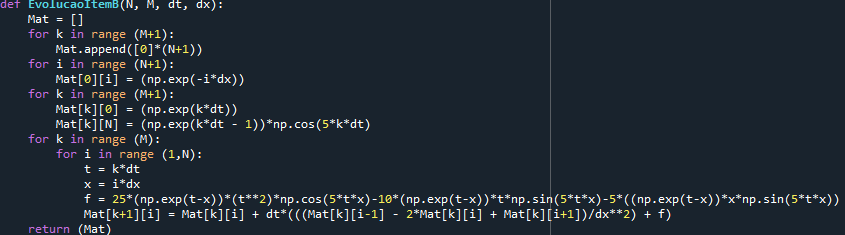
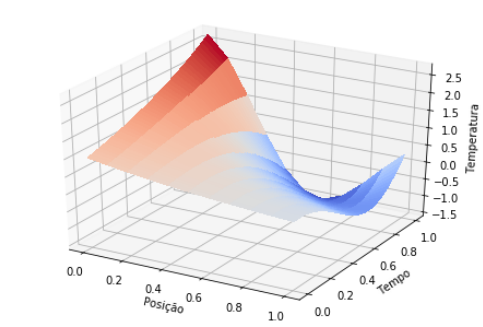
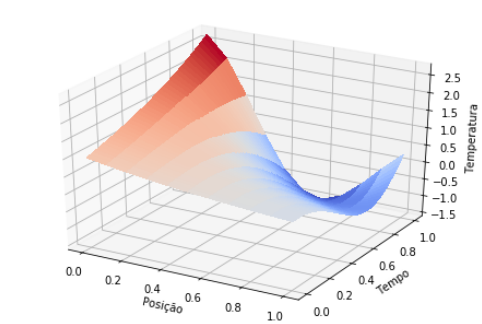
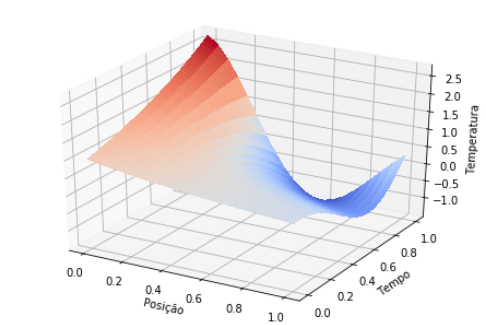
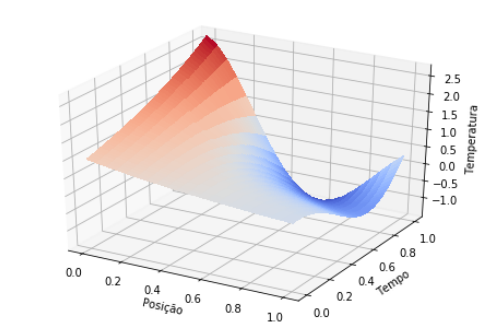
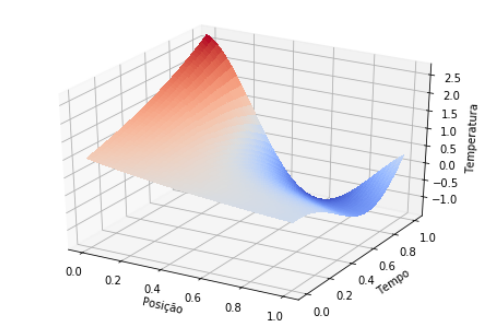
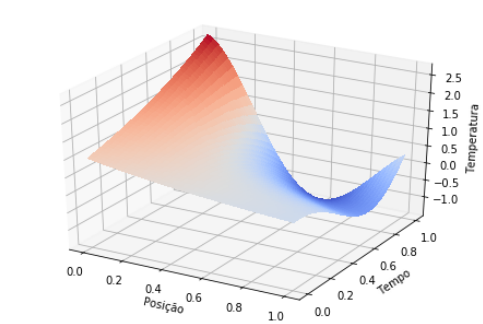
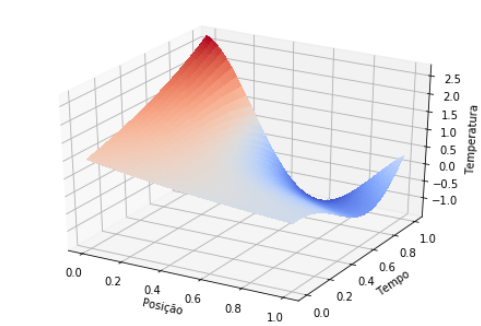
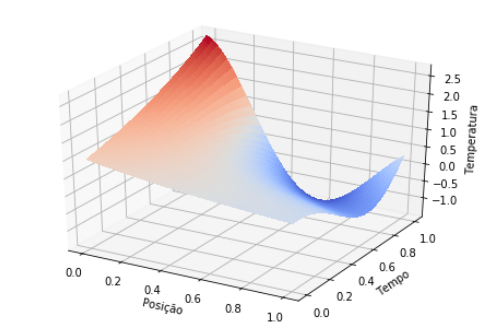
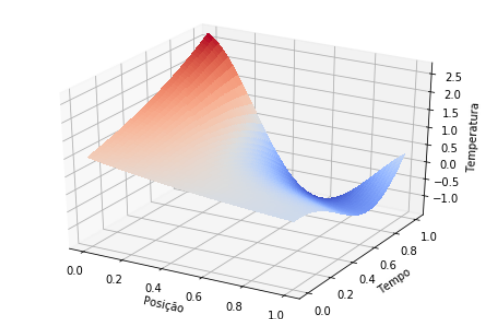
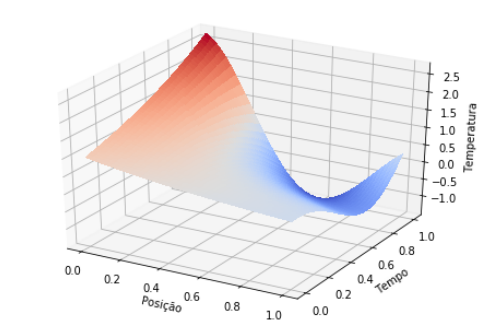
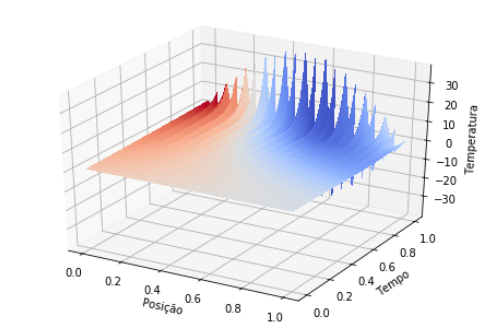
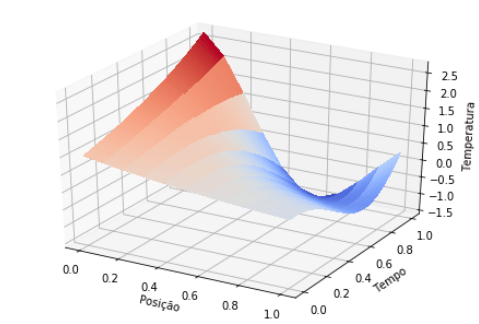
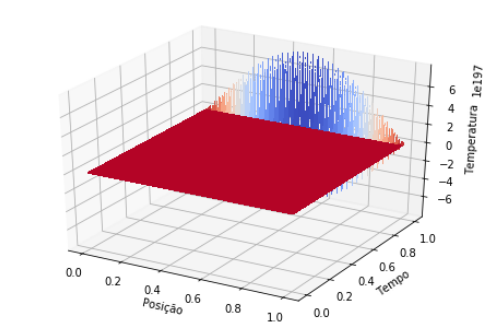
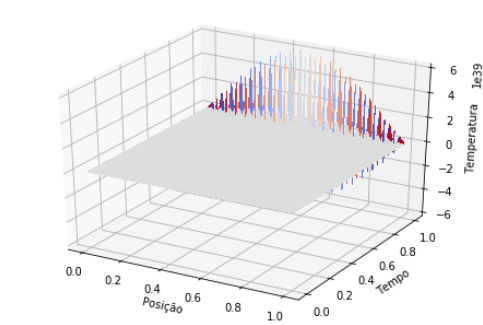
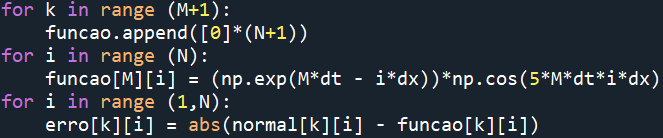
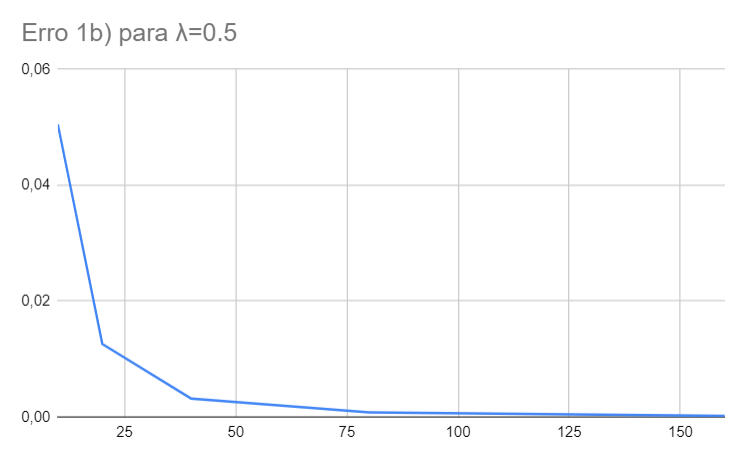
Começando pela primeira tarefa, que requisita a implementação do método 11, com valores N e M variáveis a serem escolhidas em tempo de execução.  
  
**Item a)**   
  
 No item a), faremos nosso código primeiramente determinando as condições de fronteira do gráfico, e logo apenas utilizaremos a primeira equação da parte 2 deste relatório, como na figura:  
  
 Resultando nos gráficos a seguir:

**N = 10 e λ = 0.5 N = 10 e λ = 0.25**   
  
 **N = 20 e λ = 0.5 N = 20 e λ = 0.25**  
  
  
   
 **N = 40 e** **λ = 0.5 N = 40 e λ = 0.25  
  
  
 N = 80 e** **λ = 0.5 N = 80 e λ = 0.25  
  
  
 N = 160 e** **λ = 0.5 N = 160 e λ = 0.25  
  
  
 N = 320 e** **λ = 0.5 (Obs.: devido ao tempo excessivo de execução, será a única análise com N = 320)  
** Nestas execuções, percebemos que quanto maior o N maior o tempo de execução do processo, crescendo exponencialmente. Além disso, por ambos os λ satisfazerem a condição evidenciada na introdução, o gráfico é gerado sem problemas. Quanto maior o N, também maior o leque de dados nos eixos, permitindo um gráfico mais preciso. Já para λ = 0.51:  
  
 **N = 10 e** **λ = 0.51 N = 20 e λ = 0.51**  
  
  
 **N = 40 e** **λ = 0.51 N = 80 e λ = 0.51  
** A partir desse valor de N, o programa começa a rodar com problemas, pois a solução não converge para a solução exata, deturpando o gráfico.Para convergência funcionar, ∆x deve ser da ordem de ∆t. Agora, partindo para os gráficos de erro, utilizaremos um algoritmo de comparação com a função proposta, dado a seguir:  


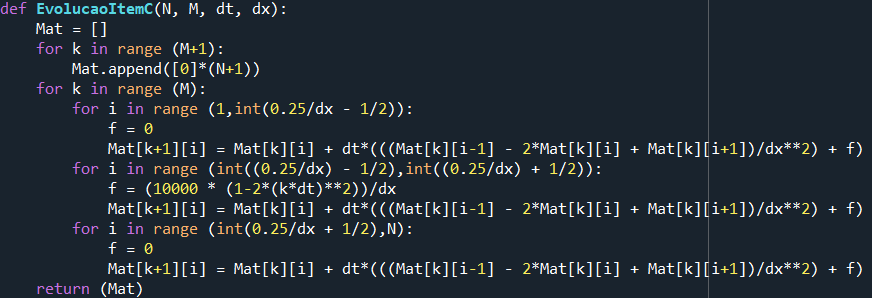
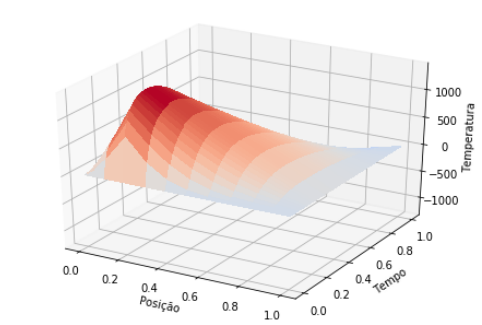
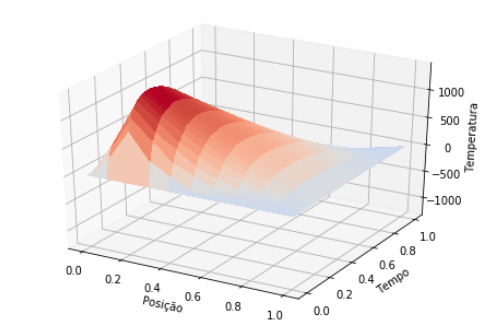
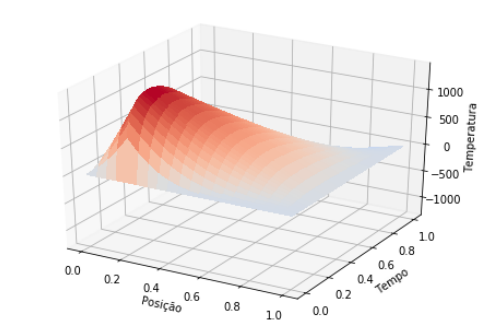
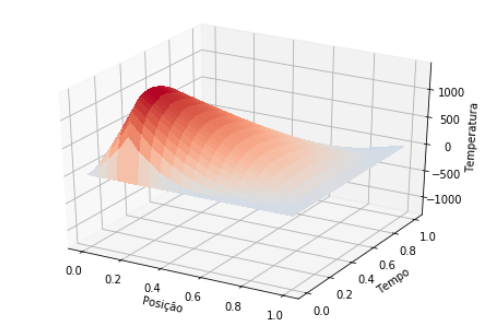
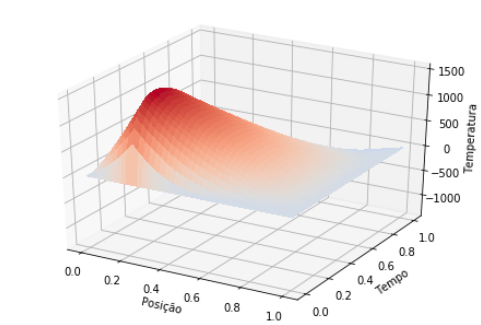
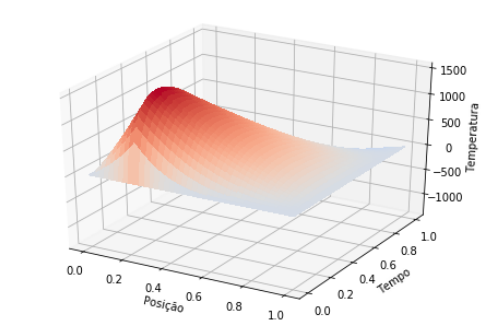
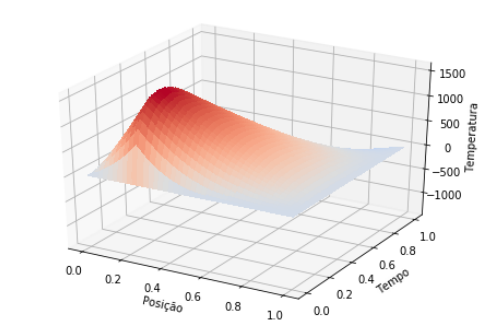
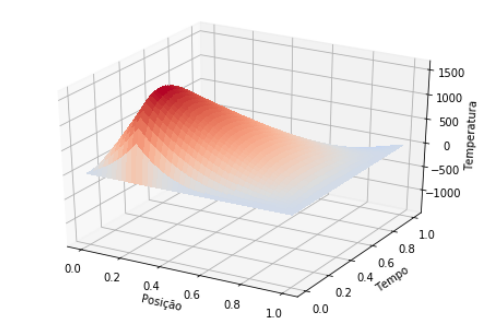
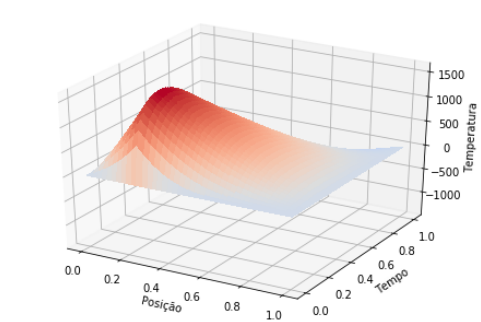
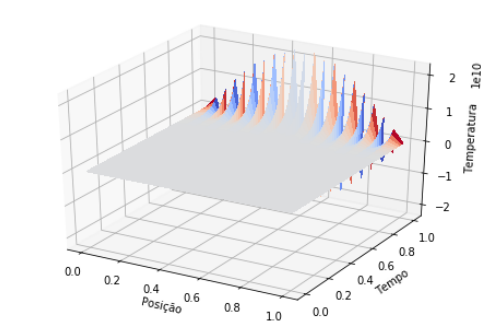
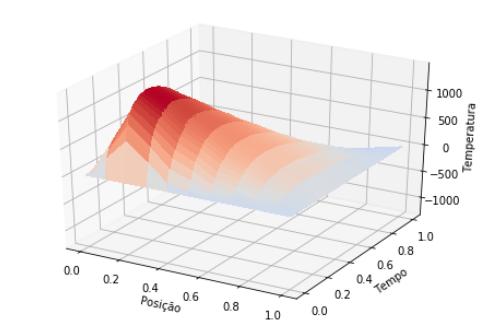
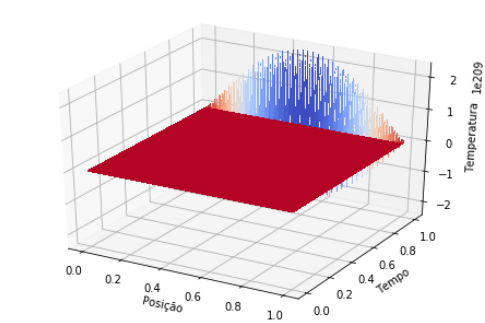
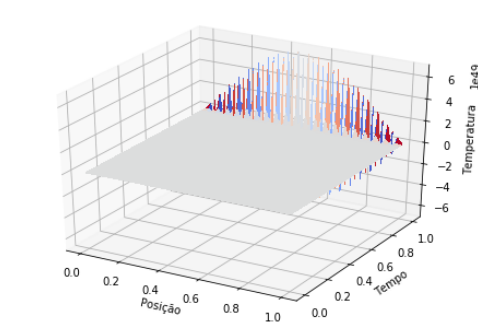
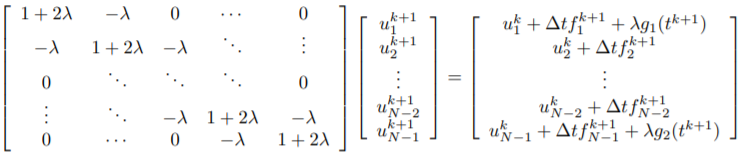
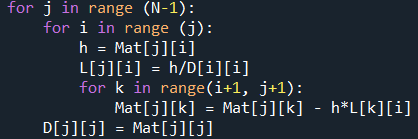
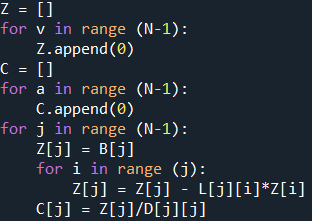
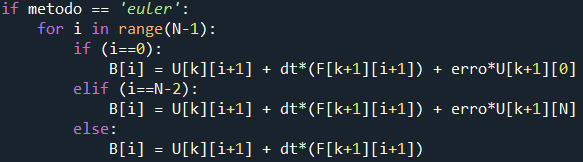
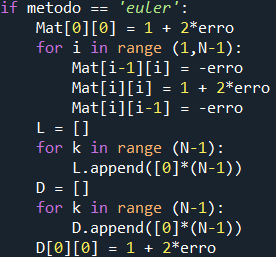
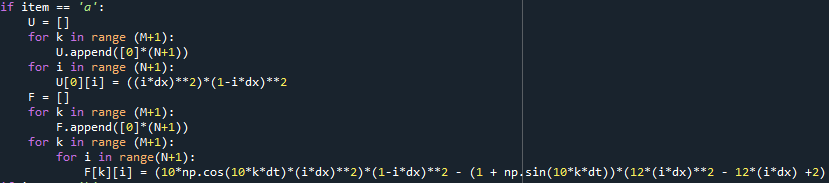
Assim, nossos gráficos de erro ficam:  
  
  
 Para o erro, quanto maior o N maior o número de pontos seguindo nossa proposta o gráfico apresenta, sendo assim, o gráfico segue para o modelo proposto quando se aumenta o N. Os valores de redução para os refinamentos de malha são apresentados a seguir:  
  
 **λ = 0.5**  **λ = 0.25**

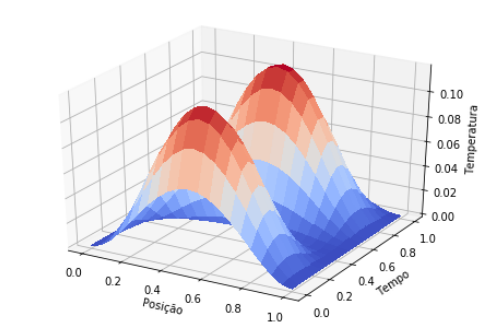
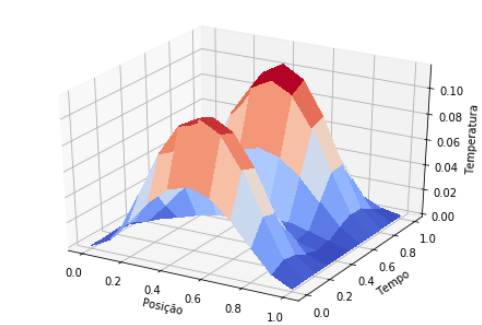
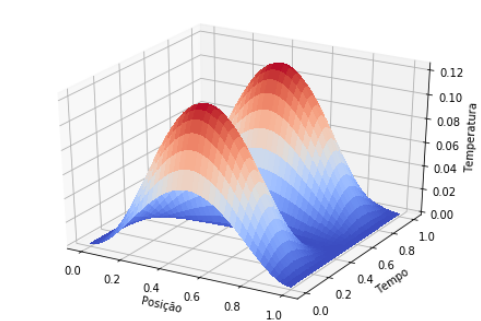
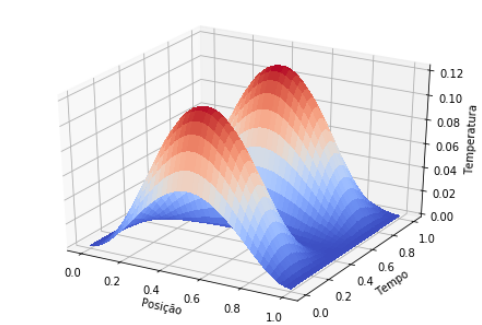
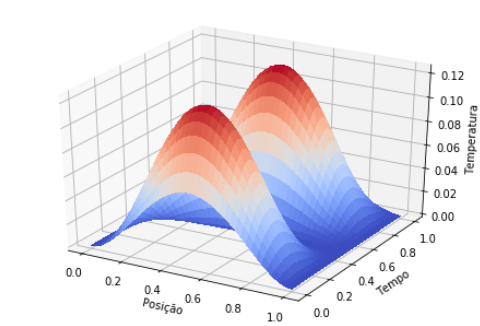
|  |  |  |
| --- | --- | --- |
| 0,2478758633 |  | 0,2487291139 |
| 0,2494710143 |  | 0,2496820699 |
| 2,49E-01 |  | 0,2494516172 |
| 2,50E-01 |  | 0,25 |

Estes valores indicam constância perante os maiores N do experimento, crescem apenas um pouco quando se eleva este valor. Quando este valor converge para 0,25 dizemos que é ordem de convergência 2.  
 Caso N fosse 640, teríamos um M de 1.638.400 (caso λ = 0.25) ou 819.200 (caso λ = 0.5), implicando em um número de passos na ordem dos milhões. Ainda se dobrássemos o N para 1280, a ordem ficaria na casa dos bilhões, sobrecarregando o processamento do computador que roda o programa.  
 Para o erro de truncamento, valores obtidos pelo programa são muito próximos de zero, podendo ser desconsiderados neste experimento computacional.  
  
**Item b)**

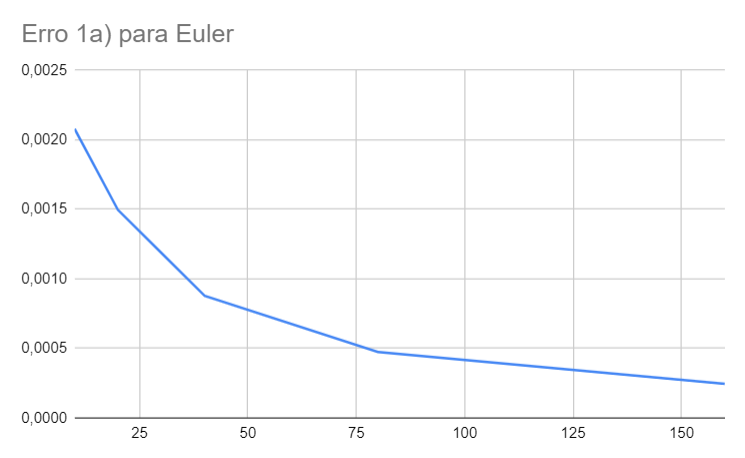
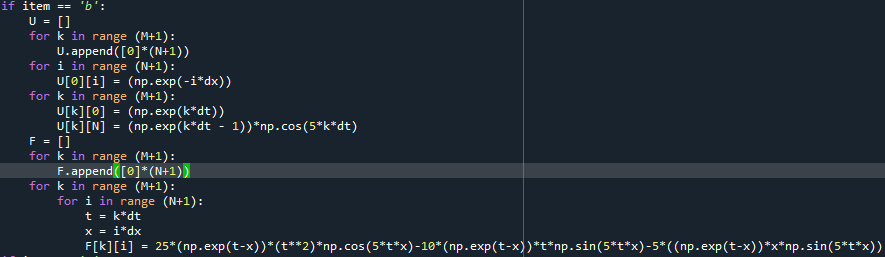
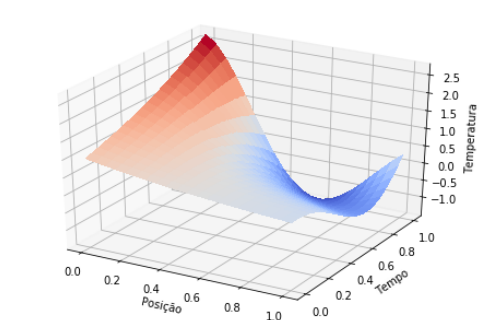
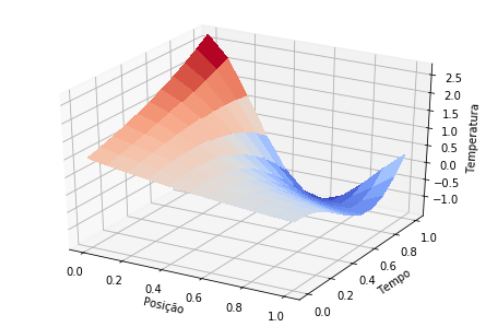
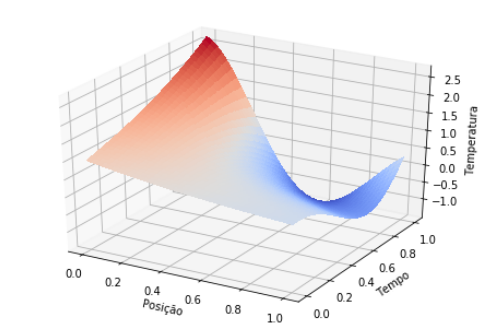
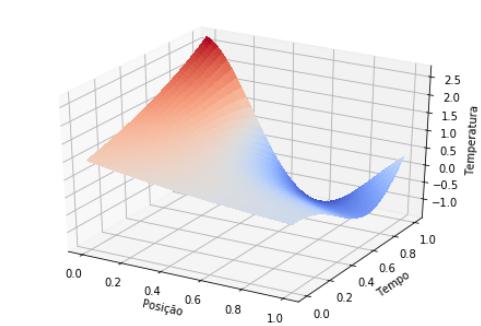
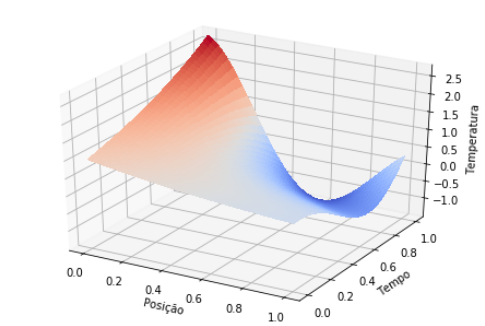
As equações solicitadas por este item estão na parte inicial deste relatório, sendo definidas por:  
ut(t, x) = uxx(t, x) + f(t, x) em [0, T] × [0, 1] -> f(t, x) = ut(t, x) - uxx(t, x)  
u(0, x) = u0(x) em [0, 1]  
u(t, 0) = g1(t) em [0, T]  
u(t, 1) = g2(t) em [0, T]  
 Sendo assim, fazendo os cálculos e dada a equação , temos que uxx(t, x) = , ut(t, x) =  (com y = x, neste caso), finalizando f(t, x) =  (sendo, novamente, y = x neste caso).  
 Já u(0, x) = u0(x) = e^(-x) / u(t, 0) = g1(t) = e^(t) / u(t, 1) = g2(t) = (e^(t-1))\*cos(5t). Com estes dados, podemos prosseguir com os resultados do programa, que será programado de forma similiar ao item a), definindo as fronteiras e depois utilizando o método:  
  
 Assim, os gráficos resultam em:  
  
 **N = 10 e** **λ = 0.5 N = 10 e λ = 0.25  
** **N = 20 e** **λ = 0.5 N = 20 e λ = 0.25  
**  
 **N = 40 e** **λ = 0.5 N = 40 e λ = 0.25  
** **N = 80 e** **λ = 0.5 N = 80 e λ = 0.25  
** **N = 160 e** **λ = 0.5 N = 160 e λ = 0.25  
** Para o item b, o resultado é similiar ao primeiro item, ao se aumentar os Ns o gráfico aparece com melhor resolução, e se aproxima da função proposta, diminuindo também seu erro que será conferido a seguir. A diferença dos lambda se torna ainda menos expressiva, quase sendo impossível de notar a diferença graficamente. Para lambda sendo 0.51:  
  
 **N = 10 e** **λ = 0.51 N = 20 e λ = 0.51  
  
  
 N = 40 e** **λ = 0.51 N = 80 e λ = 0.51  
** Semelhante ao item a, o programa constata erro com lambda 160, sendo impossível gerar um gráfico conciso. Assim, ao tentarmos rodar esse gráfico desrespeitando a premissa de lambda <= ½, percebemos que o mesmo não converge, constatando erro nos dados do mesmo. Para os gráficos de erro, mudamos apenas a função de comparação perante a utilizada no item a), porém o algoritmo é o mesmo:  
  
 Esta seleção de dados resulta em:  
O gráfico de erro se justifica semelhante, novamente, ao item a), quanto maior o arranjo de N melhor o gráfico pode ser montado próximo à função ideal. Seguindo para os valores de redução:  
 **λ = 0.5**  **λ = 0.25**

|  |  |  |
| --- | --- | --- |
| 0,2493894717 |  | 0,249747745 |
| 0,2506499504 |  | 0,2502936969 |
| 0,2499324859 |  | 0,2499166382 |
| 0,249983101 |  | 0,2500349972 |

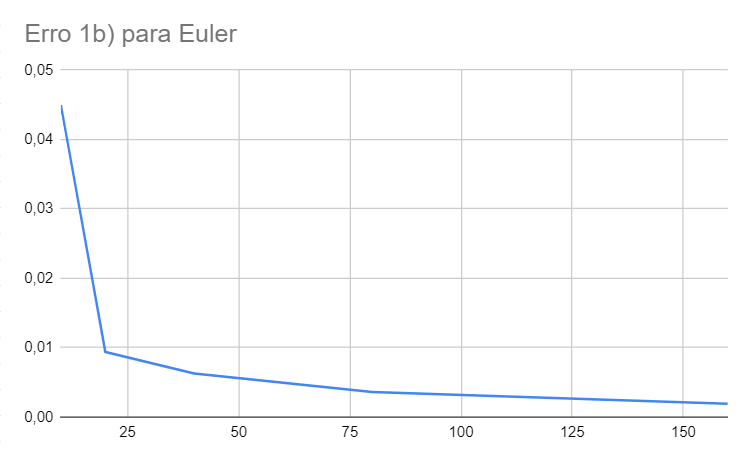
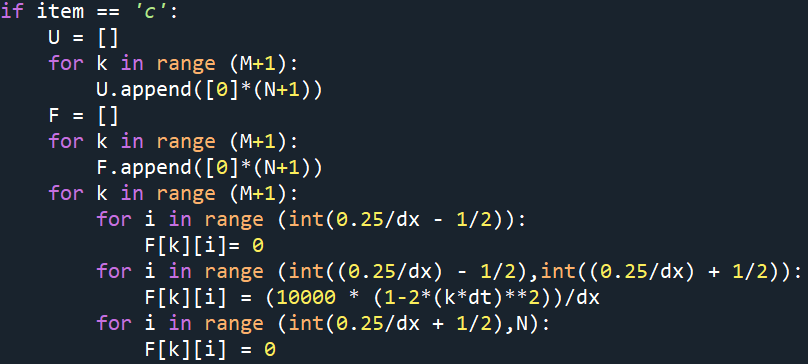
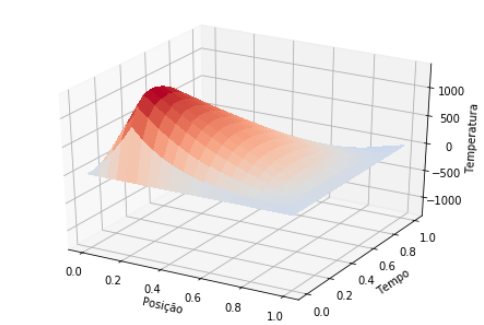
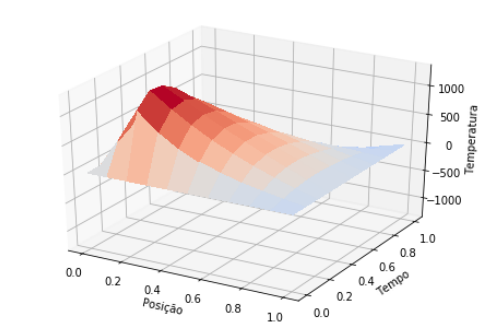
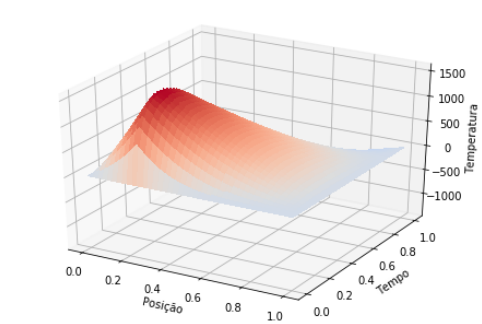
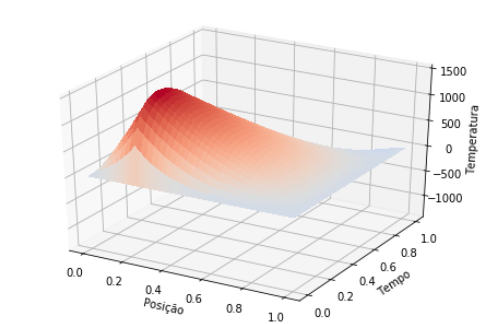
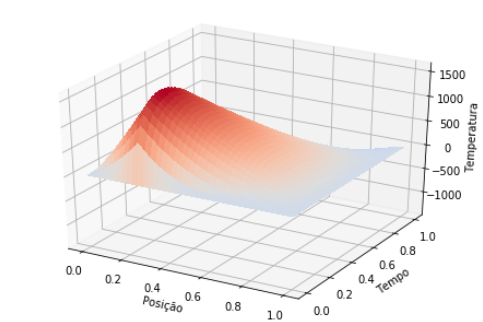
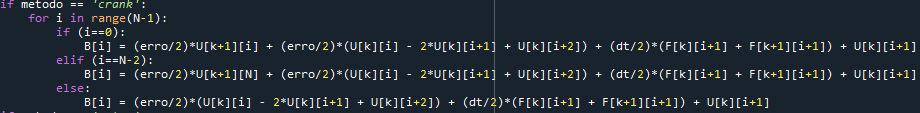
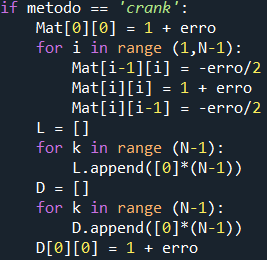
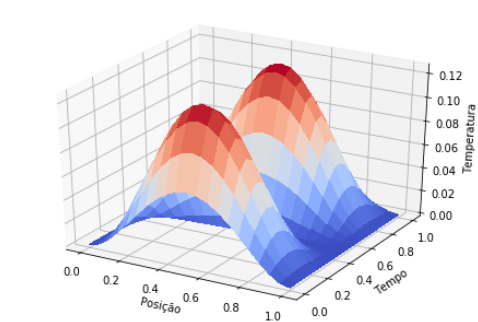
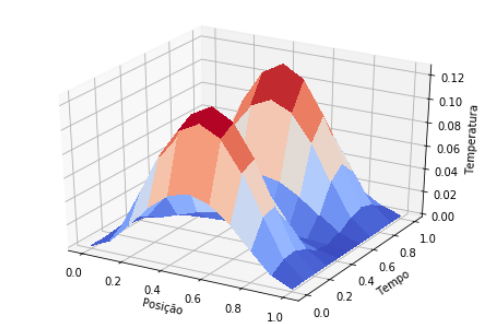
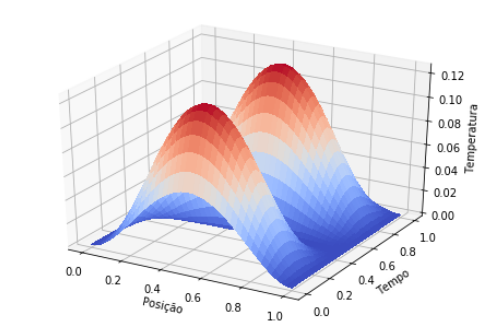
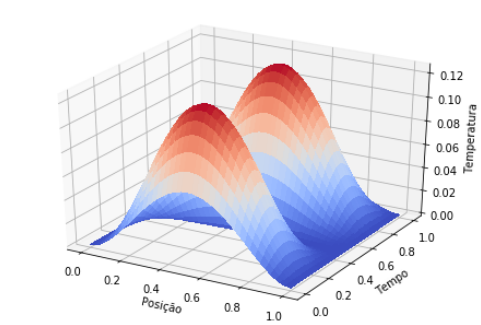
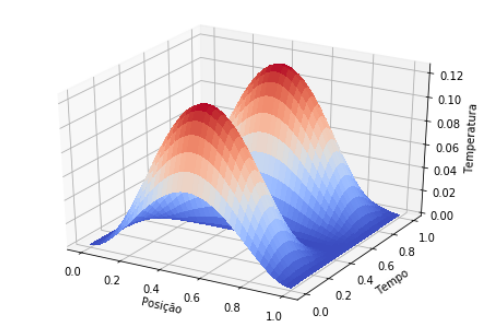
Os valores de redução se mostram semelhantes aos o item a, eles giram em torno de 0,25 com ainda menos diferenças entre os dois valores de lambda. Também se convergem para 0,25, sendo da ordem de convergência 2.  
  
**Item c)**   
  
 Neste item, devemos adotar uma equação diferente e uma proposta de resolução também diferente. A fonte pontual varia em  e as condições de contorno são diferentes perante os dados, variando em Sendo gh(x) = 1/h, definiremos a fonte pontual como f(t, x) = r(t)\*gh(x) =(10000 \* (1-2\*(t^2)))/dx, para h = ∆x. O método foi, novamente, programado similar aos outros dois porém agora como todas condições de contorno são zero, apenas foi feito um método para montar a matriz de dados:  
  
 Vamos para os gráficos:  
  
 **N = 10 e** **λ = 0.5 N = 10 e λ = 0.25  
** **N = 20 e** **λ = 0.5 N = 20 e λ = 0.25  
** **N = 40 e** **λ = 0.5 N = 40 e λ = 0.25  
** **N = 80 e** **λ = 0.5 N = 80 e λ = 0.25  
** **N = 160 e** **λ = 0.5 N = 160 e λ = 0.25  
** Os gráficos novamente se mostram semelhantes ao itens anteriores nos quesitos: diferença entre os lambda, e diferença entre os N. O formato do gráfico muda pois a função dada inicialmente mudou também, porém não muitas diferenças entre os lambda, e quanto maior os N mais preciso o gráfico. Para lambda = 0.51:  
  
 **N = 10 e** **λ = 0.51 N = 20 e λ = 0.51  
  
 N = 40 e** **λ = 0.51 N = 80 e λ = 0.51  
** O gráfico comete, novamente, os mesmo erros ao ser plotado para 0.51, evidenciando, novamente, o erro matemático em se colocar um lambda maior que 0.5.  
  
**4. Novo método de resolução** Para os próximos exercícios, utilizaremos uma resolução matricial diferente das anteriores. O método de Euler implícito é constituído pela seguinte equação () e pela forma matricial, apresentando esta forma:   
  
 Além do método de Euler, também utilizaremos o método de Crank-Nicolson, da forma: . Neste caso, trocaremos lambda por lambda/2 na forma matricial, que será protagonista na resolução exercida pelo exercício programa.  
  
**5. Tarefa 2  
  
Item a)**   
 No início desta tarefa, nos é proposto resolver por um sistema apresentado em Algebra Linear como sistema de diagonalização, pois neste é possível realizarmos operações matriciais de maneira mais simples. Sendo assim, um método de resolução foi padronizado ao longo dos três itens, de maneira que seja possível resolver todos de maneira parecida. O algoritmo primeiro transformará a matriz base em três, sendo uma delas diagonal (A = LDLt) por este método:   
.  
 Dado a matriz Mat, ele cria L e D. Após esse procedimento, serão criadas 3 matrizes de maneira que essas sirvam de transição para obtermos a matriz X solicitada pelo enunciado (). O método sera:   
.Assim, a fórmula final para resolução do exercício consta na obtenção do X (). Com todas as matrizes criadas, poderemos facilmente conseguir todos o resultados desejados.  
  
**Item b)**   
  
 Neste caso, nos é solicitado a resolução da primeira tarefa pelo método de Euler, identificado na parte 4 deste relatório. Sendo assim, o algoritmo utilizado será:  
   
  
 Ambos os algoritmos seguem o padrão proposto no método de Euler. Além disso, também foi necessário uma diferenciação pelos itens, portanto para o item a):  


Além disso, neste código não é necessário inputar lambda e também mudaremos o cálculo do M, pois é declarado no enunciado que ∆t = ∆x - > T/M = 1/N -> N = M e também lambda = ∆t/((∆x)^2) = N.  
 Com estes métodos de resolução, podemos partir aos gráficos:  
  
 **N = 10 N = 20**  
  
  
  
 **N = 40 N = 80**  
  
  
 **N = 160   
** Sendo assim, p gráfico de erro e os fatores de redução do item a) pelo método de euler são:

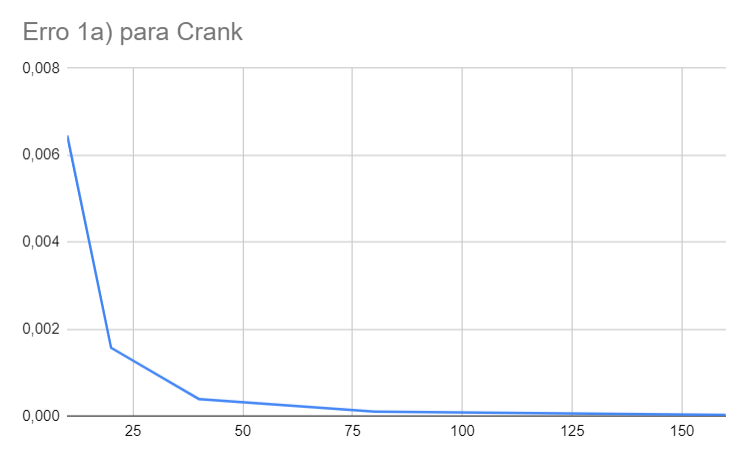
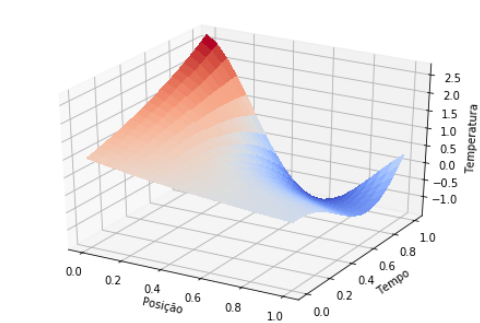
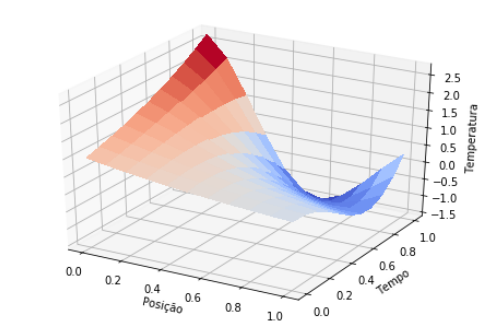
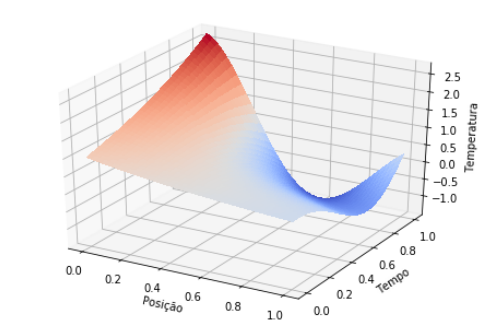
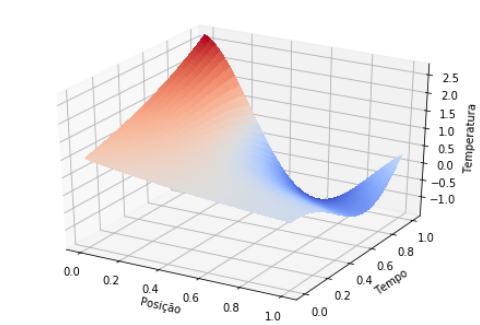
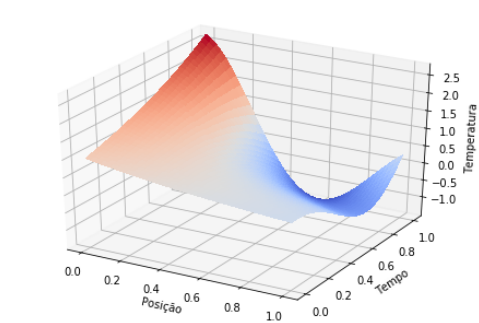
|  |
| --- |
| 0,7195791089 |
| 0,5872953749 |
| 0,5391984580 |
| 0,5185943298 |

  
  
 De acordo com os dados, podemos assumir que sua ordem de convergência é 1, pois converge para 0,5.Assim, poderemos partir para o item 1b), onde utilizamos o algoritmo:  
  
 Os gráficos para o método de Euler do item 1b) ficam:  
  
 **N = 10 N = 20  
** **N = 40 N = 80  
** **N = 160   
** Já o gráfico de erro e fator de redução do mesmo serão:

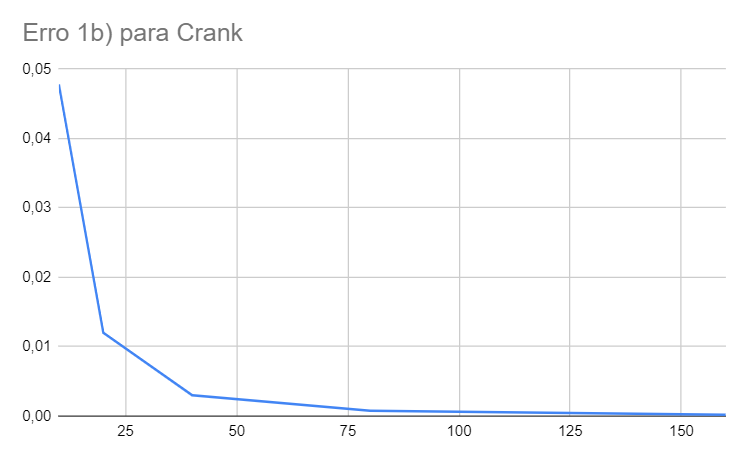
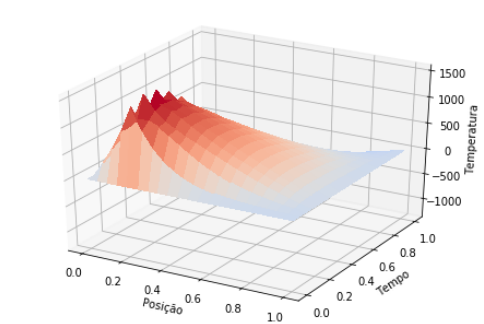
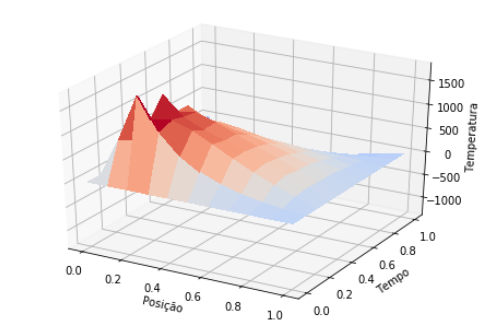
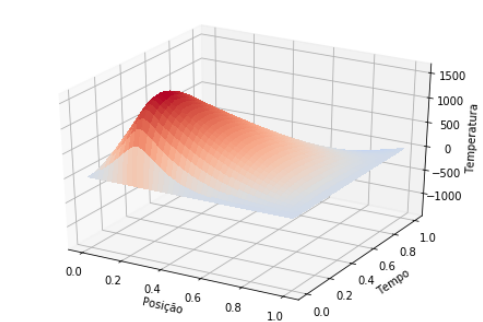
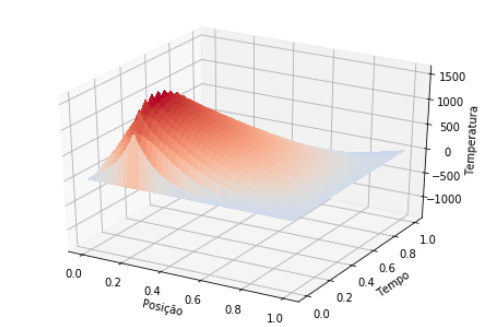
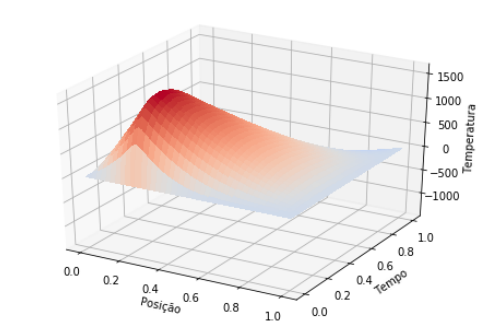
|  |
| --- |
| 0,2084039743 |
| 0,6696427779 |
| 0,5751431145 |
| 0,5345130656 |

  
 Com isso, podemos assumir a ordem de convergência 1 para estes dados. Os algoritmos utilizados foram:  
  
Assim, vamos ao item 1c):  
  
 **N = 10 N = 20**  
  
  
 **N = 40 N = 80  
** **N = 160   
** Com estes estudos, podemos finalizar o método de euler para nossos experimentos, concluindo a precisão desse método para resolução dos problemas propostos.  
  
**Item c)** Neste estudo, é proposto fazermos o mesmo do item anterior, porém agora com o método de Crank-Nicolson. Este método foi avaliado na parte 4 deste mesmo relatório, e será resolvido pelos seguintes métodos no programa:  
  
  
 Para os itens a seguir, utilizamos os mesmo métodos do item anterior, representados nas imagens de cada item. Para finalizar, vamo às análises do item 1a) por Crank:  
  
 **N = 10 N = 20  
** **N = 40 N = 80  
** **N = 160   
** O gráfico de erro e os valores de redução deste item estão apresentados a seguir:

|  |
| --- |
| 0,2430774739 |
| 0,2482071009 |
| 0,2495491621 |
| 0,2498871171 |

  
  
 Estes valores evidenciam um fator de redução tendendo a zero, portanto de ordem 2 para convergência. Assim, vamos para o item b):  
  
 **N = 10 N = 20  
** **N = 40 N = 80  
** **N = 160   
** O gráfico de erro e os valores de redução são os seguintes:

|  |
| --- |
| 0,2509458993 |
| 0,2495968556 |
| 0,2501955610 |
| 0,2499756991 |

  
  
 Os valores de redução tem convergência de grau 2. Vamos, então, para o item c):  
  
 **N = 10 N = 20**  
  
  
 **N = 40 N = 80  
** **N = 160   
** O método de Crank-Nicolson apresenta erro para N anteriores a 80, gerando um gráfico “espetado”, com erro nos valores da fronteira. Sendo assim, o método de Crank não é confiável para o caso do item c).