Método simplificado de cálculo da potência térmica resultante de cargas térmicas de refrigeração de produtos em frigoríficos domésticos

Isabel Cristina Branco Martins

Relatório do projecto final do MIEM

Orientador na FEUP: Professor Clito F. A. Afonso

Faculdade de Engenharia da Universidade do Porto

Mestrado integrado em Engenharia Mecânica

Julho de 2008

RESUMO

A preocupação com um desenvolvimento sustentável tem levado à constante investigação de processos mais eficientes na perspectiva do consumo energético. A indústria da refrigeração tem nesta procura um papel importante, uma vez que tem um peso considerável no consumo de energia eléctrica, sendo, a nível familiar, o frigorífico doméstico a aplicação responsável pelo maior consumo de energia eléctrica de uma casa. Por esta razão, é desejável a constante evolução destes electrodomésticos e consequente redução de consumo de electricidade, para, no seu pequeno contributo em cada casa, apoiar consideravelmente a protecção global do ambiente. Este trabalho foi executado com o objectivo de conhecer as cargas térmicas existentes num frigorífico e subsequente sintetização deste conhecimento, de forma simplificada, num programa de cálculo, que permitisse a determinação das cargas e a distribuição de temperaturas no frigorífico, de forma simples e rápida. Com esta análise simplificada pretende-se ajudar na escolha de componentes adequadamente dimensionados às solicitações térmicas existentes, evitando gastos energéticos desnecessários.

O programa foi feito com a base em MATLAB, sendo a simulação do frigorífico e evolução das suas condições no tempo feita através de uma interacção entre o MATLAB e um programa de simulação dinâmica, o FLUENT. No programa de cálculo criado foram analisadas as contribuições para as cargas térmicas das infiltrações, das renovações de ar aquando da abertura das portas do frigorífico, das cargas trocadas pela envolvente e as paredes do frigorífico, e as cargas introduzidas pelos produtos perecíveis, tendo sido, para os últimos, consideradas as cargas devidas ao diferencial de temperaturas e devidas à respiração e transpiração dos referidos produtos. Após o cálculo e simulação das alterações no frigorífico durante o tempo definido pelo utilizador, o programa oferece a possibilidade da visualização de variados resultados através de gráficos.

Da realização de diversos testes, em diversas condições de carregamento do frigorífico, pode-se confirmar a estabilidade do programa e repitibilidade de resultados, quando são simuladas condições idênticas em testes diferentes. Os resultados apresentados são, também, satisfatórios e concordam com valores encontrados em publicações especializadas.

ABSTRACT

The concern for sustainable development has led to constant research to find more energy efficient processes. The refrigeration industry has an important role in this demand, since it has a considerable weight in the consumption of electricity, and, at family level, the domestic refrigerator is the appliance responsible for the highest electricity consumption of a house. For this reason the constant evolution of these appliances and consequent reduction of electricity consumption is a goal, for in their small contribution in each house, they considerably contribute to the protection of the global environment. This work was carried out in order to determine the existing thermal loads in a refrigerator and subsequently synthesize this knowledge, in a simplified form, as a calculation program that allows the determination of loads and the knowledge of the distribution of temperatures in a refrigerator, simply and quickly. With this brief analysis, it is intended to help in the choice of components that provides the appropriate thermal demand, thus avoiding unnecessary energy spending.

The program was made in MATLAB, and the dynamic simulation of the refrigerator was made through an interaction between MATLAB and a CFD software, FLUENT. The program was designed to calculate the contributions to the refrigerator's thermal loads by infiltrations, renewal of air when doors are opened, due to heat transfer through the walls of the refrigerator, and loads released by perishable goods. For the latter were considered the loads due to temperature differentials and due to respiration and transpiration of some products. After the calculation and simulation of the refrigerator's loads, for the period set by the user, the program offers the possibility to view various results through graphics.

After the completion of several tests in various conditions of loading in the refrigerator, the program's stability and repeatability of results was confirmed. The results presented are also satisfactory and agree with what would should be expected.

AGRADECIMENTOS

Nesta oportunidade quero agradecer a todos os que ajudaram a completar este projecto. Primeiro, ao meu professor e orientador de projecto, Professor Clito Afonso, por me ter ajudado sempre que precisei e orientado para as outras ajudas, quando era necessário, mas, principalmente, por ter criado um projecto que me deu imenso prazer, e trabalho, a fazer.

Queria agradecer ao Professor Manuel Dias de Castro, pois, embora não fosse o meu orientador neste projecto, também agiu e me ajudou como se o fosse. Agradecimentos são também devidos aos Engenheiros José Facão e António Ferreira, pelo seu tempo e partilha de conhecimentos em FLUENT e MATLAB, respectivamente.

Agradecimentos mais pessoais devo à minha família, por nunca me ter pressionado com assuntos sobre o curso, deixando-me percorrer o meu caminho, apoiando onde necessário e ao meu namorado, pela paciência e encorajamento nas horas em que há maior desespero.

Gostaria de agradecer também aos meus colegas de trabalho, pelas tantas vezes que me mandaram estudar e pelas inúmeras horas que deram do seu trabalho para que eu pudesse concluir o meu.

Por fim, gostaria de me agradecer a mim, por não ter deixado este percurso, embora por vezes tenha sido difícil manter-me em curso.

ÍNDICE

ÍNDICE DE ILUSTRAÇÕES

ÍNDICE DE TABELAS

1. INTRODUÇÃO

No contexto económico e ambiental actual, uma das maiores preocupações e focos de desenvolvimento e investigação prende-se com a energia e sua utilização racional, reduzindo os gastos o máximo possível através da melhoria da performance dos equipamentos. Esta evolução é apenas possível com um detalhado conhecimento das necessidades que os equipamentos têm que preencher. O frigorífico doméstico é o equipamento responsável pelo maior consumo energético numa habitação e é uma comodidade que já está generalizada nos países desenvolvidos, estando cada vez mais disseminada a nível mundial. Por esta razão, os frigoríficos domésticos são alvo da atenção das empresas, que tentam melhorar os seus níveis de consumo energético. Este programa pretende ser uma ferramenta que ajuda na determinação das necessidades energéticas destes equipamentos, ajudando, através do cálculo das cargas energéticas e da simulação do frigorífico a conhecer possíveis picos de utilização de potência e também os seus regimes normais de uso, sem necessidade de uma grande experimentação, para a posterior escolha dos componentes adequados.

O conhecimento das evoluções das temperaturas no interior do frigorífico é também importante, pois, em frigoríficos sem ventilação forçada, a distribuição de temperaturas nas diversas localizações é bastante heterogénea, podendo, se não se conhecer a distribuição geral destas variações, correr o risco de estragar os produtos perecíveis guardados.

Este projecto consistia no cálculo simplificado de potências térmicas resultantes de cargas térmicas de refrigeração de produtos perecíveis em frigoríficos domésticos. Para este cálculo foi escrito um programa em MATLAB que consiste na sintetização dos conhecimentos actuais da refrigeração de produtos e transferência de calor entre superfícies e fluidos. A escolha do MATLAB para base do projecto deveu-se à extrema flexibilidade apresentada por este programa e na excelente capacidade de combinar uma potente ferramenta de cálculo com a possibilidade da criação de uma interface gráfica de fácil utilização e rápido processamento. O outro programa utilizado, o FLUENT, foi também seleccionado por ser uma das mais potentes e simples ferramentas de simulação de transferência de calor, apresentando ambos uma outra vantagem, a de poderem ser corridos através de linhas de comando, essencial para a automatização de tarefas.

No primeiro GUI (Graphical User Interface) são definidas as variáveis relativas à estrutura do frigorífico, como o seu volume, existência ou não de um congelador e o estado das borrachas das suas portas, são também definidas as temperaturas interiores e exterior e é apresentado o menu de selecção dos locais onde se vão colocar, ou se encontram já, os produtos. As informações de entrada, como a temperatura ambiente e a de refrigeração, podem ser alteradas pelo utilizador de forma simples, mas vêm já com valores pré-definidos, para facilidade de uso.

É também neste GUI que é colocado o botão de cálculo final, onde vão ser realizados os cálculos necessários à apresentação dos resultados desejados. Um botão ‘Gráficos’ abre um novo menu, onde o utilizador pode escolher gráficos de evolução das cargas térmicas ou de temperaturas, ao longo da simulação. Dentro destas categorias, o utilizador pode ainda seleccionar o visionamento das cargas ou temperaturas por cada local do frigorífico, por cada tipo de produtos, por cada produto individual, as cargas

por respiração dos produtos, as cargas por infiltrações e por renovações de ar, e também pode visionar a evolução das temperaturas no ar do frigorífico como um todo.

Nos sub-menus dos tipos de produtos em cada local do frigorífico são dadas listas de produtos que o utilizador pode seleccionar, escolhendo a quantidade desse produto, indicando, em alguns casos, a unidade dessa quantidade, e seleccionando a temperatura a que o produto se encontra.

A evolução do projecto pode-se dividir em cinco fases.

A fase inicial consistiu na compilação de informações relativas às propriedades térmicas dos produtos perecíveis utilizados habitualmente que requerem refrigeração, em publicações especializadas e através da internet. Foram também recolhidas informações acerca dos processos que ocorriam na refrigeração dos produtos e no interior do frigorífico, na transferência de calor entre as superfícies dos produtos e das paredes com o ar interior através da condução, convecção natural e radiação.

Numa segunda fase do projecto foram definidos os objectivos do programa, ou seja, que resultados iria apresentar e foi esboçado o caminho e recursos que seriam necessários utilizar para concretizar os objectivos. Foi estudado o funcionamento do programa e linguagem de programação MATLAB após a conclusão de que este seria o programa mais indicado para este projecto. Em seguida foi planeada a aparência e distribuição de informações das interfaces com o utilizador, com a atenção em tornar a sua interpretação e utilização o mais simples possível, mantendo todas as funcionalidades desejadas.

Numa terceira fase, após terem sido criados os menus de interface com o utilizador iniciais, foi iniciada a programação em MATLAB, criando as ligações internas entre os menus, criando as funções e procedimentos ligados aos cálculos que seriam necessários efectuar e ligando os menus às respectivas matrizes de informação e funções. Foi feito um esforço para que todos os possíveis erros derivados da utilização do programa por um utilizador inexperiente fossem comunicados e resolvidos, para continuar com a correcta simulação das condições pretendidas.

A quarta fase prendeu-se com a necessidade de simular o interior do frigorífico, sendo a simulação completa em MATLAB impraticável, uma vez que se teria que criar um completo programa de simulação de fluidos em regime transitório, algo que não é nativo ao MATLAB. Por esta razão foi escolhido outro programa para a realização desta simulação, e foi iniciada a aprendizagem da utilização do programa FLUENT e sua possível programação para automatização de procedimentos de simulação.

Numa quinta e última fase de projecto foram ligadas as funções pretendidas do MATLAB e do FLUENT, para que os seus cálculos se complementassem e o programa oferecesse resultados de uma simulação de qualidade, numa perspectiva de utilizações simples, uma vez que se trata do cálculo simplificado de cargas térmicas.

Os resultados obtidos com este programa foram, inicialmente, sujeitos a uma análise crítica de senso comum, sendo observadas as evoluções das temperaturas e do calor libertado pelos produtos perecíveis no interior do frigorífico. Após ter sido refinado o programa as vezes necessárias até esta análise ser positiva, foram comparados os valores obtidos com alguns valores encontrados na literatura especializada,

nomeadamente em publicações da ASHRAE. Nesta comparação, a análise revelou-se também positiva, embora nem todos os resultados coincidam com os publicados, estão próximos deles. A não concordância pode-se dever, em grande parte, ao facto de os resultados publicados serem resultados experimentais, enquanto que os obtidos pelo programa são fruto de algumas simplificações e arredondamentos que se propagam ao longo dos cálculos. No entanto, como foi mencionado, os resultados obtidos estão bastante próximos deles, pelo que se pode afirmar que o programa simula as condições dadas correctamente. Também foram realizados testes de repetição de condições de carregamento do frigorífico para verificar se as mesmas condições em simulações distintas originavam resultados iguais. Este teste foi superado, indicando que o programa era estável e consistente nos seus cálculos. O programa foi também testado a nível de facilidade de utilização, sendo usado por utilizadores que não conheciam o seu funcionamento, mas que conseguiram facilmente compreender as informações que eram requeridas como entradas e como obter os resultados que queriam analisar como saídas.

2. DETERMINAÇÃO DAS PROPRIEDADES TÉRMICAS NECESSÁRIAS À MODELAÇÃO DOS PRODUTOS PERECÍVEIS

Para a modelação destes produtos foram criadas matrizes em MATLAB com os valores das percentagens dos seus elementos constituintes, água, proteínas, gordura, hidratos de carbono, fibras e cinzas. Estes valores foram retirados das publicações ASHRAE Refrigeration; 2002 (1) e "Viva melhor"; 2006(2). A lista de produtos apresentados não é exaustiva e, em alguns casos, não apresenta produtos tipicamente portugueses, devido à inexistência ou dificultada acessibilidade a resultados de estudos desta natureza dos produtos nacionais. Foram também criadas listagens dos pesos individuais e dimensões dos produtos, recorrendo ao sítio www.continente.pt e a visitas ao hipermercado em questão.

Após terem sido recolhidos estes dados, foram definidas fórmulas que calculassem as propriedades térmicas de cada produto, recorrendo à formulação que se segue. O método utilizado para o cálculo das propriedades dos produtos é o indicado pela ASHRAE Refrigeration; 2002 (1).

Para constituintes alimentares:

- condutividade térmica (W/m.K):

Tabela 1: Condutividade térmica

proteínas:	$k = 1{,}7881\times10^{-1} + 1{,}1958\times10^{-3}\cdot T - 2{,}7178\times10^{-6}\cdot T^2$
gorduras:	$k = 1{,}8071\times10^{-1} - 2{,}7604\times10^{-3}\cdot T - 1{,}7749\times10^{-7}\cdot T^2$
hidratos de carbono:	$k = 2{,}0141\times10^{-1} + 1{,}3874\times10^{-3}\cdot T - 4{,}3312\times10^{-6}\cdot T^2$
fibras:	$k = 1{,}8331\times10^{-1} + 1{,}2497\times10^{-3}\cdot T - 3{,}1683\times10^{-6}\cdot T^2$
cinzas:	$k = 3{,}2962\times10^{-1} + 1{,}4011\times10^{-3}\cdot T - 2{,}9069\times10^{-6}\cdot T^2$

- difusividade térmica (m^2/s):

Tabela 2: Difusividade térmica

proteínas:	$\alpha = 6{,}8714\times10^{-8} + 4{,}7578\times10^{-10}\cdot T - 1{,}4646\times10^{-12}\cdot T^2$
gorduras:	$\alpha = 9{,}8777\times10^{-8} - 1{,}2569\times10^{-10}\cdot T - 3{,}8286\times10^{-14}\cdot T^2$
hidratos de carbono:	$\alpha = 8{,}0842\times10^{-8} + 5{,}3052\times10^{-10}\cdot T - 2{,}3218\times10^{-12}\cdot T^2$
fibras:	$\alpha = 7{,}3976\times10^{-8} + 5{,}1902\times10^{-10}\cdot T - 2{,}2202\times10^{-12}\cdot T^2$
cinzas:	$\alpha = 1{,}2461\times10^{-7} + 3{,}7321\times10^{-10}\cdot T - 1{,}2244\times10^{-12}\cdot T^2$

- massa volúmica (kg/m^3):

Tabela 3: Massa volúmica

proteínas:	$\rho = 1{,}3299\times10^{3} - 5{,}1840\times10^{-1}\cdot T$
gorduras:	$\rho = 9{,}2559\times10^{2} - 4{,}1757\times10^{-1}\cdot T$
hidratos de carbono:	$\rho = 1{,}5991\times10^{3} - 3{,}1046\times10^{-1}\cdot T$
fibras:	$\rho = 1{,}3299\times10^{3} - 5{,}1840\times10^{-1}\cdot T$
cinzas:	$\rho = 2{,}4238\times10^{3} - 2{,}8063\times10^{-1}\cdot T$

- calor específico (kJ/kg.K):

Tabela 4: Calor específico

proteínas:	$c_p = 2{,}0082 + 1{,}2089\times10^{-3}\cdot T - 1{,}3129\times10^{-6}\cdot T^2$
gorduras:	$c_p = 1{,}9842 + 1{,}4733\times10^{-3}\cdot T - 4{,}8008\times10^{-6}\cdot T^2$
hidratos de carbono:	$c_p = 1{,}5488 + 1{,}9625\times10^{-3}\cdot T - 5{,}9399\times10^{-6}\cdot T^2$
fibras:	$c_p = 1{,}8459 + 1{,}8306\times10^{-3}\cdot T - 4{,}6509\times10^{-6}\cdot T^2$
cinzas:	$c_p = 1{,}0926 + 1{,}8896\times10^{-3}\cdot T - 3{,}6817\times10^{-6}\cdot T^2$

Para água e gelo:

- condutividade térmica (W/m.K):

Tabela 5: Condutividade térmica da água

água:	$k = 5{,}7109\times10^{-1} + 1{,}7625\times10^{-3}\cdot T - 6{,}7036\times10^{-6}\cdot T^2$
gelo:	$k = 2{,}2196 - 6{,}2489\times10^{-3}\cdot T + 1{,}0154\times10^{-4}\cdot T^2$

- difusividade térmica (m^2/s):

Tabela 6: Difusividade da água

água:	$\alpha = 1{,}3168\times10^{-7} + 6{,}2477\times10^{-10}\cdot T - 2{,}4022\times10^{-12}\cdot T^2$
gelo:	$\alpha = 1{,}1756\times10^{-6} - 6{,}0833\times10^{-9}\cdot T + 9{,}5037\times10^{-11}\cdot T^2$

- massa volúmica (kg/m^3):

Tabela 7: Massa volúmica da água

água:	$\rho = 9{,}9718\times10^{2} + 3{,}1439\times10^{-3}\cdot T - 3{,}7574\times10^{-3}\cdot T^2$
gelo:	$\rho = 9{,}1689\times10^{2} - 1{,}3071\times10^{-1}\cdot T$

- calor específico (kJ/kg.K):

Tabela 8: Calor específico da água

água dos -40 ºC aos 0 ºC:	$c_p = 4{,}0817 - 5{,}3062\times10^{-3}\cdot T + 9{,}9516\times10^{-4}\cdot T^2$
água dos 0 ºC aos 150 ºC:	$c_p = 4{,}1762 - 9{,}0864\times10^{-5}\cdot T + 5{,}4731\times10^{-6}\cdot T^2$
gelo:	$c_p = 2{,}0623 + 6{,}0769\times10^{-3}\cdot T$

Sendo T a temperatura do produto a que se quer avaliar a propriedade térmica, em ^{o}C.

Uma outra parte essencial da modelação dos produtos consistia na determinação das cargas emitidas por produtos vivos, como os vegetais e os frutos, através da sua respiração, uma vez que estas cargas variam com o tipo de produtos estudados e a sua temperatura. Com base na tabela apresentada abaixo, retirada da ASHRAE Refrigeration; 2002 [(1)], foram obtidas relações entre as temperaturas dos produtos e a sua taxa de respiração, através de regressões polinomiais, potenciais, ou exponenciais, dependendo da que melhor aproximasse a correlação em temperaturas mais baixas, entre os 0^{o}C e os 20^{o}C. Como não existiam estudos para todos os produtos indicados no programa, apenas os que estavam tabelados são considerados na contribuição do calor libertado por respiração para o ar do frigorífico.

Tabela 9: Potência de respiração

Potência de respiração (mW/kg)						
Produto:	0^{o}C	5^{o}C	10^{o}C	15^{o}C	20^{o}C	25^{o}C
maçãs	6.8-12.1	15-21.3		40.3-91.7	50-103.8	
damascos	15.5-17	18.9-26.7	33-55.8	63-101.8	87.3-155.2	
alcachofras	67.4-133.4	94.6-178	16.2-291.5	22.9-430.2	40.4-692	
espargos	81-237.6	162-404.5	318.1-904	472.3-971.4	809.4-1484	
abacates				183.3-465.6	218.7-1029.1	
bananas				37.3-164.9	97-242.5	
feijões		101.4-103.8	162-172.6	252.2-276.4	350.6-386	
feijões-lima	31-89.2	58.2-106.7		296.8-369.5	393.8-531.5	
beterrabas	16-21.3	27.2-28.1	34.9-40.3	50-68.9		
amoras silvestres	46.6-67.9	84.9-135.8	155.2-281.3	208.5-431.6	388-581.9	
arandos	6.8-31	27.2-36.4		101.4-183.3	153.7-259	
amoras		12.1-13.6			32.5-53.8	
framboesas	52.4-74.2	91.7-114.4	82.4-164.9	243.9-300.7	339.5-727.4	

morangos	36.4-52.4	48.5-98.4	145.5-281.3	210.5-273.5	303.1-581	501.4-625.6
bróculos	55.3-63.5	102.3-474.8		515-1008.2	824.9-1011.1	1155.2-1661
couves de Bruxelas	45.6-71.3	95.5-144	187.2-250.7	283.2-316.7	267.2-564	
repolho	28.1-40.35	52.4-63.5	86.3-98.4	159.1-167.7		
cenouras	10.2-20.4	17.5-35.9	29.1-46.1	86.8-196.4 (a 18ºC)		
couve-flor	22.8-71.3	58.2-81	121.2-144.5	199.8-243		
aipo	15-21.3	27.2-37.8	58.2-81	115.9-124.1 (a 18ºC)		
cerejas	12.1-16	28.1-41.7		74.2-133.4	83.4-94.6	
milho	126.1	230.4	332.2	483	855.5	1207.5
pepinos			68.4-85.8 (a 13ºC)	71.3-98.4	92.1-142.6	
figos		23.5-39.3	65.5-68.4	145.5-187.7	168.8-281.8	252.2-281.8
alhos	8.7-32.5	17.5-28.6	27.2-28.6	32.5-81	29.6-53.8	
uvas	3.9-6.8	9.2-17.5	2.42	29.6-34.9		74.2-89.2
toranjas				37.8	47	56.7
rábano silvestre	24.2	32	78.1	97	132.4	
kiwis	8.3	19.6	38.9		51.9-57.3	
alhos-porros	28.1-48.5	58.2-86.3	159.1-202.2	245.4-346.7		
limões				47	67.4	77.1
alface		61.6	105.2	131.4	203.2	321.5
limas			7.8-17	17.5-31	20.4-55.3	44.6-134.8
mangas				133.4	222.6-449.1	356
meloas		25.7-29.6	46.1	99.9-114.4	132.4-191.6	184.8-211.9
melões			23.8	34.9-47	59.2-70.8	78.1-102.3

melancias			22.3		51.4-74.2	
cogumelos	83.4-129.5	210.5			782.2-938.9	
quiabo			259	432.6	774.5	1024 (a 29ºC)
azeitonas				64.5-115.9	114.4-145.5	121.2-180.9
cebolas	8.7	10.2	21.3	33	50	83.4 (a 29ºC)
laranjas	9.2	18.9	36.4	62.1	89.2	105.2 (a 27ºC)
salsa	98-136.5	195.9-252.3	288.8-486.7	427.4-661.9	581.7-756.8	914.1-1012
pêssegos	12.1-18.9	18.9-27.2		98.4-125.6	175.6-303.6	241.5-361.3
peras	7.8-14.5	21.8-46.1	21.9-63	101.8-160	116.4-166.7	
ervilhas	90.2-138.7	163.4-226.5		530.1-600.4	728.4-1072.2	1018.4-1118.3
pimentos			42.7	67.9	130	
dióspiros		17.5		34.9-41.7	59.2-71.3	86.3-118.8
ananáses			22.3	53.8	118.3	185.7
ameixas	5.8-8.7	11.6-26.7	26.7-33.9	35.4-36.9	53.3-77.1	82.9-210.5
batatas		17.5-20.4	19.7-29.6	19.7-34.9	19.7-47	
rábanos	16-17.5	22.8-24.2	44.6-97	82.4-97	141.6-145.5	199.8-225.5
ruibarbos	24.2-39.3	32.5-53.8		91.7-134.8	118.8-168.8	
espinafres	34.3-63.5	81-95.5	173.6-222.6		549-641.6	
abobrinhas			103.8-109.1	222.6-269.6	252.2-288.6	
batatas doces				84.9		160.5-217.3
tomates verdes				60.6	102.8	126.6 (a 27ºC)
tomates maduros				79.1	120.3	143.1 (a 27ºC)

As correlações obtidas são as que se seguem, em mW/kg.

Frutos:

Tabela 10: Correlações da potência de respiração para os frutos

maçãs:	c_resp=0.073*temp_p^2+0.977*temp_p+0.16
damascos:	c_resp=0.165*temp_p^2+0.456*temp_p+14.24
abacates:	c_resp=107.9*exp(0.035*temp_p)
bananas:	c_resp=2.12*exp(0.191*temp_p)
amoras silvestres:	c_resp=0.062*temp_p^3-1.125*temp_p^2+14.46*temp_p+43.85
arandos:	c_resp=0.212*temp_p^2+3.106*temp_p+6.65
cerejas:	c_resp=-0.045*temp_p^2+4.68*temp_p+10.01
amoras:	c_resp=8.704*exp(0.065*temp_p))
figos:	c_resp=2.26*temp_p^1.472
toranjas:	c_resp=20.68*exp(0.04*temp_p)
uvas:	c_resp=-0.001*temp_p^3+0.216*temp_p^2-1.492*temp_p+5.494
limões:	c_resp=23.22*exp(0.049*temp_p)
limas:	c_resp=2.851*exp(0.107*temp_p)
mangas:	c_resp=30.81*exp(0.098*temp_p)
melões:	c_resp=0.078*temp_p^2+1.014*temp_p+4.93
azeitonas:	c_resp=2.201*temp_p^1.27
laranjas:	c_resp=-0.008*temp_p^3+0.369*temp_p^2-0.068*temp_p+9.666
pêssegos:	c_resp=0.33*temp_p^2+1.237*temp_p+9.017
pêras:	c_resp=8.424*exp(0.138*temp_p)
dióspiros:	c_resp=0.159*temp_p^2-1.307*temp_p+19.84
ananás:	c_resp=0.097*temp_p^2.351
ameixas:	c_resp=0.004*temp_p^3-0.07*temp_p^2+2.134*temp_p+4.973
framboesas:	c_resp=49.38*exp(0.094*temp_p)
morangos:	c_resp=0.652*temp_p^2+1.711*temp_p+36.66
meloas:	c_resp=-0.009*temp_p^3+0.527*temp_p^2-0.88*temp_p+16.68
melancias:	c_resp=9.674*exp(0.083*temp_p

Vegetais:

Tabela 11: Correlações da potência de respiração para os vegetais

alcachofras:	c_resp=-0.017*temp_p^4+0.784*temp_p^3-10.78*temp_p^2+41.96*temp_p+67.4
espargos:	c_resp=1.716*temp_p^2-2.676*temp_p+38.93
feijões:	c_resp=-0.028*temp_p^3+1.448*temp_p^2-4.606*temp_p+91.8
feijões lima:	c_resp=31.47*exp(0.134*temp_p)
beterrabas:	c_resp=0.014*temp_p^3-0.288*temp_p^2+3.316*temp_p+16
bróculos:	c_resp=-0.041*temp_p^3+2.977*temp_p^2-4.657+55.58
couves de Bruxelas:	c_resp=-0.102*temp_p^3+2.711*temp_p^2-2.141*temp_p+46.72
cenouras:	c_resp=0.055*temp_p^3-0.75*temp_p^2+3.816*temp_p+10.2
couves-flor:	c_resp=0.432*temp_p^2+5.4*temp_p+22.2
aipos vermelhos:	c_resp=0.01*temp_p^3+0.218*temp_p^2+1.086*temp_p+15
aipos:	c_resp=0.01*temp_p^3+0.218*temp_p^2+1.086*temp_p+15
milho:	c_resp=0.031*temp_p^3+0.468*temp_p^2+11.88*temp_p+134.4
pepinos:	c_resp=37.77*exp(0.044*temp_p)
alhos:	c_resp=-0.006*temp_p^3+0.102*temp_p^2+1.42*temp_p+8.678
rábanos silvestres:	c_resp=-0.014*temp_p^3+0.516*temp_p^2+0.903*temp_p+22.64
alhos-porros:	c_resp=0.562*temp_p^2+6.626*temp_p+23.83
alfaces:	c_resp=42.92*exp(0.079*temp_p)
cogumelos:	c_resp=0.634*temp_p^2+22.24*temp_p+83.4
quiabo:	c_resp=105.1*exp(0.094*temp_p)
cebolas:	c_resp=-8e-05*temp_p^3+0.097*temp_p^2+0.199*temp_p+8.246
salsa:	c_resp=0.09*temp_p^3-2.677*temp_p^2+43.06*temp_p+83.98
ervilhas:	c_resp=1.722*temp_p^2+10.28*temp_p+140.2
pimentos:	c_resp=13.59*exp(0.111*temp_p)
batatas:	c_resp=-0.022*temp_p^2+0.682*temp_p+14.75
rábanos:	c_resp=-0.004*temp_p^3+0.467*temp_p^2-1.375*temp_p+16.7
ruibarbos:	c_resp=0.125*temp_p^2+2.461*temp_p+21.82
espinafres:	c_resp=1.716*temp_p^2-2.676*temp_p+38.93
abobrinhas:	c_resp=5.308*temp_p^1.32
tomates	c_resp=2.177*temp_p^1.249

maduros verdes:	
tomates maduros:	c_resp=5.429*temp_p^1.005

Os produtos, além de libertarem calor através da respiração, libertam também calor sensível se existir um diferencial de temperaturas entre estes e o ar que os rodeia. Esse calor foi calculado através das expressões indicadas em seguida, retiradas da publicação de R.N. Ballard; ASHRAE Transactions[(3)].

1 – Calor libertado da temperatura do produto até uma temperatura acima da de congelamento:

$$Q_1 = m \cdot cp \cdot (T_i - T_f)$$

Q_1 – energia retirada do produto (kJ);

m – massa do produto (kg);

C_p – calor específico do produto acima da temperatura de congelamento (kJ/kg.K);

T_i – temperatura inicial do produto (K ou ^oC);

T_f – temperatura acima da de congelamento do produto (K ou ^oC).

Para o refrigerador, apenas este passo é necessário, sendo a temperatura T_f a do refrigerador.

2 – Calor libertado durante o congelamento:

$$Q_2 = m \cdot h_{if}$$

Q_2 – energia retirada do produto (kJ);

m – massa do produto (kg);

h_{if} – calor latente de fusão do produto (kJ/kg).

3 – Calor libertado da temperatura de congelamento até à temperatura final (temperatura de câmara de congelamento):

$$Q_3 = m \cdot cp \cdot (T_z - T_f)$$

Q_3 – energia retirada do produto (kJ);

m – massa do produto (kg);

C_p – calor específico do produto abaixo da temperatura de congelamento (kJ/kg.K);

T_z – temperatura de congelamento do produto (K ou ^oC);

T_f – temperatura final do produto (K ou ^oC).

A carga libertada por diferencial de temperaturas total é dada por:

$$Q_{total} = Q_1 + Q_2 + Q_3$$

A potência libertada por cada produto é a soma do calor libertado pelo diferencial de temperaturas e pela potência libertada por respiração e por transpiração.

O tempo de arrefecimento de cada produto é calculado no instante inicial, através do procedimento indicado pela ASHRAE Refrigeration; 2002 (1). O processo é descrito em seguida.

1 – Tempo de refrigeração:

1º - determinar as propriedades térmicas do produto;

2º - determinar o coeficiente de transferência de calor à superfície, para o processo de refrigeração;

3º - determinar a dimensão característica L e os rácios dimensionais b_1 e b_2;

$$\beta_1 = \frac{2^a\, menor\ dimensão\ do\ produto}{menor\ dimensão\ do\ produto}$$

$$\beta_2 = \frac{maior\ dimensão\ do\ produto}{menor\ dimensão\ do\ produto}$$

4º - calcular o número de Biot;

$$Bi = \frac{h \cdot L}{k}$$

h – coeficiente de transferência de calor por convecção ($W/m^2.K$);

L – dimensão característica do produto (m);

k – condutividade térmica do produto (W/m.K);

5º - calcular a dimensionalidade equivalente da transferência de calor E para a geometria do produto, conhecendo E_0 e E_∞;

$$E = \frac{Bi^{\frac{4}{3}} + 1{,}85}{\frac{Bi^{\frac{4}{3}}}{E_\infty} + \frac{1{,}85}{E_0}}$$

Para produtos bidimensionais de forma irregular (E igual para Bi=0):

$$E_0 = \left(1 + \frac{1}{\beta_1}\right) \cdot \left[1 + \left(\frac{\beta_1 - 1}{2 \cdot \beta_1 + 2}\right)^2\right]$$

Para produtos tridimensionais de forma irregular:

$$E_0 = 1{,}5 \cdot \frac{\beta_1 + \beta_2 + {\beta_1}^2 \cdot (1 + \beta_2) + {\beta_2}^2 \cdot (1 + \beta_1)}{\beta_1 \cdot \beta_2 \cdot (1 + \beta_1 + \beta_2)} - \frac{[(\beta_1 - \beta_2)^2]^{0,4}}{15}$$

Para cilindros finitos, blocos e troncos rectangulares infinitos:

$$E_0 = 1 + \frac{1}{\beta_1} + \frac{1}{\beta_2}$$

Para esferas:

$$E_0 = 3$$

Para cilindros infinitos:

$$E_0 = 2$$

Para placas infinitas:

$$E_0 = 1$$

Para produtos bi e tridimensionais, E_0 quando Bi $\rightarrow \infty$:

$$f(\beta) = \frac{1}{\beta^2} + 0{,}01 \cdot p_3 \cdot exp\left(\beta - \frac{\beta^2}{6}\right)$$

onde.

p_1, p_2, p_3 – parâmetros geométricos;

6º - calcular o factor de retardação, j_m correspondente ao centro térmico ou ao centro mássico do produto;

$$j_c = \frac{Bi^{1,35} + \frac{1}{\lambda}}{\frac{Bi^{1,35}}{L_\infty} + \frac{1}{\lambda}}$$

λ, γ e N - valores tabelados;

$$L_\infty = 1{,}271 + 0{,}305 \cdot exp(0{,}172 \cdot \gamma_1 - 0{,}115 \cdot {\gamma_1}^2) + 0{,}425 \cdot exp(0{,}09 \cdot \gamma_2 - 0{,}128 \cdot {\gamma_2}^2)$$

$$j_m = \mu \cdot j_c$$

(j_m para a temperatura média da massa)

$$\mu = \left(\frac{1{,}5 + 0{,}69 \cdot Bi}{1{,}5 + Bi}\right)^N$$

7º - Calcular a raiz da equação transcendental (em radianos);

$$\omega \cdot cot(\omega) + Bi - 1 = 0$$

8º - calcular o tempo de refrigeração do produto.

$$\theta = \frac{3 \cdot \rho \cdot c \cdot L^2}{\omega^2 \cdot k \cdot E} \cdot ln\left(\frac{j}{Y}\right)$$

e,

$$Y = \frac{T_m - T}{T_m - T_i}$$

θ – tempo de refrigeração (s);

ρ – massa volúmica do produto (kg/m^3);

c – calor específico do produto (J/kg.K);

L – raio de metade da grossura do produto (m);

k – condutividade térmica do produto (W/m.K);

j – factor de retardação;

E – dimensionalidade equivalente da transferência de calor do produto;

ω – primeira raiz, em radianos, da equação transcendental;

T_m – temperatura do meio de refrigeração (K ou ^{o}C);

T – temperatura do produto (K ou ^{o}C);

T_i – temperatura inicial do produto (K ou ^{o}C);

2 – Tempo de congelamento:

Este método é válido para produtos de diversas geometrias, como troncos rectangulares infinitos, cilindros finitos, blocos rectangulares tridimensionais e geometrias irregulares bi e tridimensionais;

1º - determinar as propriedades térmicas do produto;

2º - determinar o coeficiente de transferência de calor à superfície, para o processo de congelamento;

3º - determinar a dimensão característica D do produto e os rácios dimensionais β_1 e β_2;

$$\beta_1 = \frac{2^a\, menor\ dimensão\ do\ produto}{menor\ dimensão\ do\ produto}$$

$$\beta_2 = \frac{maior\ dimensão\ do\ produto}{menor\ dimensão\ do\ produto}$$

4º - calcular o número de Biot, o número de Plank e o número de Stefan;

$$Bi = \frac{h \cdot D}{k}$$

h – coeficiente de transferência de calor por convecção (W/m^2.K);

D – dimensão característica (m),

k – condutividade térmica (W/m.K);

$$Pk = \frac{C_l \cdot (T_i - T_f)}{\Delta H}$$

C_l – calor específico volúmico da fase não-congelada (kJ/m^3.K);

ΔH – variação da entalpia volumétrica entre T_i e T_f (kJ/m^3);

T_i – temperatura inicial do produto (K ou ^{o}C);

T_f – temperatura final do produto (K ou ^{o}C);

$$Ste = \frac{C_s \cdot (T_f - T_m)}{\Delta H}$$

C_s – calor específico volúmico da fase congelada (kJ/m^3.K);

T_m – temperatura de congelamento (K ou ^{o}C);

ΔH – variação da entalpia volumétrica entre T_f e T_m (kJ/m^3);

5º - cálculo do tempo de refrigeração de uma placa infinita através de um método apropriado;

$$\theta = \frac{\Delta H_{18}}{\Delta T} \cdot \left(\frac{P \cdot D}{h} + \frac{R \cdot D^2}{k_s}\right) \cdot \left[1 - \frac{1{,}65 \cdot Ste}{k_s} \cdot ln\left(\frac{T_c - T_m}{T_{ref} - T_m}\right)\right]$$

$$\Delta T = (T_f - T_m) + \frac{(T_i - T_f)^2 \cdot \frac{C_l}{2} - (T_f - T_c)^2 \cdot \frac{C_s}{2}}{\Delta H_{18}}$$

T_c – temperatura final do centro do produto (K ou ^{o}C);

ΔH – variação da entalpia volumétrica entre a temperatura inicial e a temperatura final do centro do produto, assumida -18ºC;

$$P = o,7306 - 1{,}083 \cdot Pk + Ste \cdot \left(15{,}40 \cdot U - 15{,}43 + 0{,}01329 \cdot \frac{Ste}{Bi}\right)$$

$$R = 0{,}2079 - 0{,}2656 \cdot U(Ste)$$

e,

$$U = \frac{\Delta T}{T_f - T_m}$$

6º - calcular a dimensionalidade equivalente da transferência de calor para a geometria do produto;

7º - calcular o tempo de congelamento do produto.

$$\theta_{forma} = \frac{\theta_{placa}}{E}$$

Para blocos rectangulares com dimensões D por b_{11}D por b_2D, a dimensionalidade equivalente de transferência de calor é:

$$E = 1 + W_1 + W_2$$

$$W_1 = \left(\frac{Bi}{Bi+2}\right) \cdot \frac{5}{8 \cdot {\beta_1}^3} + \left(\frac{2}{Bi+2}\right) \cdot \frac{2}{\beta_1 \cdot (\beta_1 + 1)}$$

$$W_2 = \left(\frac{Bi}{Bi+2}\right) \cdot \frac{5}{8 \cdot {\beta_2}^3} + \left(\frac{2}{Bi+2}\right) \cdot \frac{2}{\beta_2 \cdot (\beta_2 + 1)}$$

Para cilindros finitos, onde o diâmetro é menor que a altura, a dimensionalidade equivalente de transferência de calor é dada por:

$$E = 2{,}0 + W_2$$

Para placas infinitas, cilindros finitos e infinitos, blocos rectangulares, esferas e formas irregulares bi e tri-dimensionais, Cleland et al. desenvolveram as seguintes expressões de cálculo da dimensionalidade equivalente da transferência de calor:

$$E = G_1 + G_2 \cdot E_1 + G_3 \cdot E_2$$

onde,

$$E_1 = X \cdot \left(\frac{2{,}32}{{\beta_1}^{1{,}77}}\right) \cdot \frac{1}{\beta_1} + \left[1 - X \cdot \left(\frac{2{,}32}{{\beta_1}^{1{,}77}}\right)\right] \cdot \frac{0{,}73}{{\beta_1}^{2{,}50}}$$

$$E_1 = X \cdot \left(\frac{2{,}32}{{\beta_1}^{1{,}77}}\right) \cdot \frac{1}{\beta_1} + \left[1 - X \cdot \left(\frac{2{,}32}{{\beta_1}^{1{,}77}}\right)\right] \cdot \frac{0{,}73}{{\beta_1}^{2{,}50}}$$

$$X(x) = \frac{x}{Bi^{1,34} + x}$$

Com todos os elementos anteriores determinados, é possível obter a temperatura a que os produtos se encontram, em cada passo de dois minutos, através da equação:

$$\dot{Q}_{produto} = m \cdot cp \cdot \left(T_i - T_f\right) \cdot t_{arrefecimento}$$

$\dot{Q}_{produto}$– potência libertada pelo produto (W);

m – massa do produto (kg);

c_p – calor específico do produto (kJ/kg.K);

T_i – temperatura inicial do produto (K ou ^oC);

T_f – temperatura final do produto (K ou ^oC),

$t_{arrefecimento}$ – tempo de arrefecimento do produto desde a sua temperatura inicial até à temperatura da sua envolvente,

em que a potência libertada pelos produtos é actualizada para cada passo, sendo considerada a temperatura da sua envolvente a temperatura do ar que rodeia a sua localização.

Para melhor visualização, é apresentado o esquema de cálculos geral necessário para a obtenção das cargas térmicas libertadas pelos produtos.

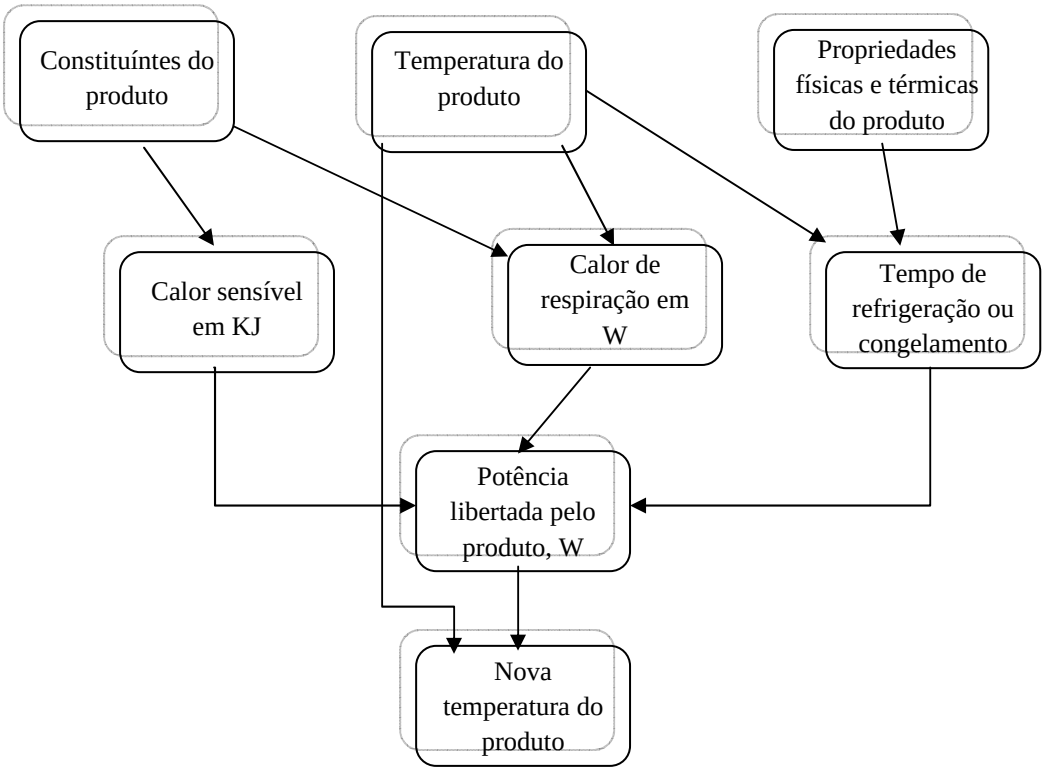


Ilustração 1: Esquema de cálculo para a obtenção das cargas térmicas dos produtos

3. MODELAÇÃO DO FRIGORÍFICO

Para a simulação do ar interior do frigorífico no FLUENT foi necessário o desenho de uma estrutura semelhante à de um frigorífico real e subsequente criação da malha dos volumes a analisar. Para a modelação do frigorífico foram comparados os modelos existentes no mercado e feita uma aproximação das suas medidas. São possíveis de modelar, com este programa, frigoríficos de 4 prateleiras, com ou sem congelador, e com volumes compreendidos entre os 200L e os 300L, ao todo, 11 capacidades do refrigerador, uma vez que a capacidade do congelador foi mantida fixa.

Foram consideradas algumas hipóteses simplificativas nesta etapa:

- foi considerado um coeficiente de transferência de calor do exterior para o interior do frigorífico constante, calculado para as temperaturas ambiente, do refrigerador e congelador escolhidas pelo utilizador;

- a temperatura exterior é constante ao longo do tempo e definida pelo utilizador;

- o evaporador, na parte do refrigerador, foi simulado como uma parede fria, em que a sua temperatura é constante e igual a -1,15^{o}C, valor médio entre as paragens e arranques do compressor, na zona do congelador, a temperatura é de -23,15^{o}C;

- os volumes ocupados pelos produtos são simplificados por cargas paralelepipédicas, com ar circulante à sua volta;

- não foram consideradas prateleiras na simulação, por se tratarem apenas de grelhas cuja influência na transferência de calor por condução é pequena, relativamente aos outros fenómenos de transferência de calor presentes,

- como não é viável, em termos de tempo, fazer uma simulação contínua, foram escolhidos períodos de simulação de 2 minutos, durante os quais se considera que os produtos mantêm uma emissão de potência calorífica constante e igual ao valor calculado para o respectivo passo,

- a temperatura do produto, aquando da sua entrada no programa pelo utilizador é considerada uniforme em todo o produto, seja a temperatura ambiente, a de refrigeração ou a de congelamento;

- a simulação inicial parte de um estado permanente, em que todos os produtos que se encontram no frigorífico estão já à temperatura da sua envolvente.

Para a determinação do coeficiente de transferência global e a potência térmica que atravessa as paredes do frigorífico foi utilizada a formulação seguinte, para cálculo dos processos de condução, entre as diversas camadas das paredes, de radiação, tanto exterior como interior, e também convecção natural no exterior e no interior, uma vez que não foi considerada a existência de ventilação forçada no interior do frigorífico

$$\dot{Q}_{envolvente} = U \cdot A \cdot (T_{ext} - T_{int})$$

onde

$$U = \frac{1}{R_{total}}$$

$\dot{Q}_{envolvente}$ - potência calorífica total trocada através da envolvente (W);

U – coeficiente de transferência de calor global ($W/m^2.K$);

A – área superficial da parede (m^2);

T_{ext} – temperatura do ar exterior (ºC ou K);

T_{int} – temperatura do ar interior (ºC ou K);

R_{total} – resistência térmica total ($m^2.K/W$);

Condução através das paredes:

$$\dot{Q}_{condução} = \frac{T_{ext} - T_{int}}{R_{condução}} \quad \text{e} \quad R_{condução} = \frac{e}{\lambda \cdot A}$$

$\dot{Q}_{condução}$ – potência calorífica que atravessa a parede (W);

T_{ext} – temperatura do ar exterior (ºC ou K);

T_{int} – temperatura do ar interior (ºC ou K);

$R_{condução}$ – resistência térmica de condução (K/W);

e – espessura da parede (m);

λ – condutibilidade térmica da parede (W/m.K);

A – área superficial da parede (m^2);

Convecção:

$$\dot{Q}_{convecção} = \frac{T_{ext} - T_{int}}{R_{convecção}} \quad \text{e} \quad R_{convecção} = \frac{1}{\alpha_{cv} \cdot A}$$

com

$$\alpha_{cv} = \frac{Nu \cdot \lambda_{ar}}{L_{ref}}$$

$\dot{Q}_{convecção}$ – potência calorífica de convecção (W);

T_{ext} – temperatura do ar exterior (ºC ou K);

T_{int} – temperatura do ar interior (ºC ou K);

$R_{convecção}$ – resistência térmica de convecção (K/W);

α_{cv} – coeficiente de convecção ($W/m^2.K$);

A – área superficial da parede (m^2);

Nu – número de Nusselt;

λ – condutibilidade térmica do ar (W/m.K);

L_{ref} – comprimento característico da superfície da parede (m);

Radiação:

$$\dot{Q}_{radiação} = \frac{T_p - T_{sup}}{R_{radiação}} \quad \text{e} \quad R_{radiação} = \frac{1}{\alpha_{rad} \cdot A}$$

com

$$\alpha_{rad} = \varepsilon \cdot \sigma \cdot \left(T_p + T_{sup}\right) \cdot \left({T_p}^2 + {T_{sup}}^2\right)$$

$\dot{Q}_{radiação}$ – potência calorífica de radiação (W);

T_p – temperatura da parede (ºC ou K);

T_{sup} – temperatura da superfície (ºC ou K);

$R_{radiação}$ – resistência térmica de radiação (K/W);

α_{rad} – coeficiente de radiação (W/m^2.K);

A – área superficial da parede (m^2);

ε - emissividade;

σ – constante de Stefan-Boltzmann ($\sigma = 5{,}729 \times 10^{-7}$W/m^2.K^4);

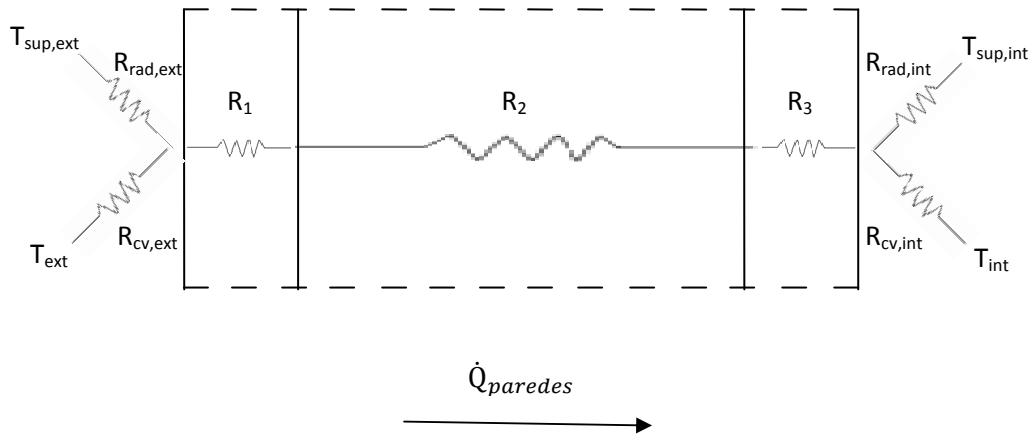


Ilustração 2: Esquema das resistências térmicas da parede

Como exemplificado no diagrama, a resistência térmica total é, para cada parede:

$$R_{total} = \frac{R_{cv,ext} \cdot R_{rad,ext}}{R_{cv,ext} + R_{rad,ext}} + R_1 + R_2 + R_3 + \frac{R_{cv,int} \cdot R_{rad,int}}{R_{cv,int} + R_{rad,int}}$$

onde R_1, R_2 e R_3 são, respectivamente, as resistências térmicas de condução do material exterior, do isolamento e do material interior.

À potência transmitida pela envolvente foram também adicionadas as potências transmitidas por infiltrações e por renovações de ar, uma vez que todas elas incidiam sobre a mesma sub-divisão do frigorífico, o seu ar interior. As infiltrações foram

calculadas de 2 em 2 minutos, o tempo de cada passo, e para as renovações de ar foi calculada a média da sua potência em 24h, com os dados fornecidos pelo utilizador, sendo depois este valor dividido para também entrar apenas, em cada passo, o contributo de 2 minutos desta potência.

Para a obtenção das cargas introduzidas no frigorífico pelas infiltrações que, após algum tempo de uso, as borrachas das portas permitem, foi utilizado como base o trabalho de pesquisa de Clito Afonso, Manuel Castro e Joaquim Matos; "Air infiltration on domestic refrigerators: the influence of magnetic seals" [3], sendo utilizada a formulação seguinte:

$$\bar{I} = \frac{1}{t} \cdot ln\left(\frac{C_0}{C}\right)$$

$\bar{I}$ – infiltração média de ar (s^{-1}):

t – tempo (minutos);

C_0 – concentração inicial do gás traçador;

C – concentração, a um tempo t, do gás traçador;

Infiltrações em selos novos:

refrigerador:

$ln(C) = -2{,}1227 \cdot t + 7{,}9683$

congelador:

$ln(C) = -1{,}1070 \cdot t + 7{,}7824$

Infiltrações em selos velhos:

refrigerador:

$ln(C) = -12{,}6610 \cdot t + 13{,}9190$

congelador:

$ln(C) = -6{,}0419 \cdot t + 11{,}0440$

O calor devido às infiltrações é dado por:

$$\dot{Q} = \dot{m} \cdot cp \cdot (T_{ext} - T_{int}) \quad \text{e} \quad \dot{m} = \dot{V} \cdot \rho = V \cdot I \cdot \rho$$

$\dot{Q}$ - potência calorífica das infiltrações (W);

$\dot{m}$ - caudal de ar infiltrado (kg/s);

c_p – calor específico do ar infiltrado (kJ/kg.K);

T_{ext} – temperatura do ar exterior (ºC ou K);

T_{int} – temperatura do ar interior (ºC ou K);

$\dot{V}$ – caudal volúmico do ar infiltrado (m^3/s);

P – massa volúmica do ar infiltrado (kg/m^3);

V – volume de ar infiltrado (m3);

I – taxa de infiltração de ar (s^{-1});

Para a obtenção das cargas térmicas devidas às renovações de ar, foi utilizada a formulação indicada pela ASHRAE Refrigeration; 2002 (1), como descrita em seguida.

$$\dot{q}_t = \dot{q} \cdot D_t \cdot D_f \cdot (1 - E)$$

$\dot{q}_t$ – carga térmica média para 24 horas (kW);

$\dot{q}$ – carga de refrigeração sensível e latente para escoamento completamente desenvolvido (kW);

D_t – factor duração do tempo de abertura da porta;

D_f – factor escoamento através da porta;

E – efectividade da protecção oferecida pela porta.

Para a carga térmica de um escoamento completamente desenvolvido, foi usada a equação desenvolvida por Gosney e Olama; 1975:

$$q = 0.221 \cdot A \cdot (h_i - h_r) \cdot \rho_r \cdot \sqrt{\left(1 - \frac{\rho_i}{\rho_r}\right)} \cdot \sqrt{g \cdot H} \cdot F_m$$

onde

$$F_m = \left(\frac{2}{1 + \left(\frac{\rho_r}{\rho_i}\right)^{\frac{1}{3}}} \right)^{1.5}$$

q – carga de refrigeração sensível e latente (kW);

A – área da porta (m^2);

h_i – entalpia do ar infiltrado (kJ/kg);

h_r – entalpia do ar refrigerado (kJ/kg);

ρi – massa volúmica do ar infiltrado (kg/m3);

ρr – massa volúmica do ar refrigerado (kg/m3);

g – aceleração da gravidade = 9,81 m/s2;

H – altura da porta (m);

Fm – factor massa volúmica.

Finalmente, para o cálculo do factor duração do tempo de abertura da porta, como se trata de uma utilização irregular, foi utilizada a expressão seguinte:

$$D_t = \frac{\left(P \cdot \theta_p + 60 \cdot \theta_o\right)}{3600 \cdot \theta_d}$$

D_t – fracção decimal do tempo de duração de abertura da porta;

P – número diário de aberturas da porta;

θ_p – tempo que demora a abrir/fechar a porta (s);

θ_o – tempo que a porta fica aberta (minutos);

θ_d – período de tempo diário=24h.

Foi definido 1 segundo para o tempo que demora a abrir ou a fechar a porta, uma vez que frigoríficos domésticos não oferecem grande resistência à abertura nem têm massas elevadas, que aumentariam o tempo de abertura. O número de vezes que as portas são abertas num dia, assim como a duração média dessas aberturas, são definidos pelo utilizador, vindo, no entanto, com valores pré-definidos de 10 aberturas diárias para o refrigerador e 3 para o congelador, com uma duração média de 15 segundos cada.

Para o factor escoamento através da porta, é sugerido por Hendrix et al; 1989, um valor de 0,8, para uma diferença de temperaturas de 16^{o}C, com portas de rápida abertura, condições que na generalidade se aplicam ao frigorífico doméstico. A efectividade da protecção oferecida pela porta é, segundo a ASHRAE, de 0,95 para equipamentos novos. Este valor não é mudado no caso de o utilizador indicar que as membranas de protecção das portas são velhas, pois a sua contribuição para o aumento da carga térmica é contabilizado nas infiltrações.

Após o cálculo de todas as cargas que envolviam a estrutura do frigorífico, a mesma foi criada com o programa GAMBIT, um software de desenho CAD/CAE que exporta as estruturas e malhas criadas para diversos programas de análise, entre eles, o utilizado neste projecto, o FLUENT. Foi desenhado o interior do frigorífico, e, uma vez que foram englobadas as cargas que envolviam a estrutura física, ou seja, as paredes e portas do frigorífico, estas não foram desenhadas, tendo sido apenas representado o ar interior do frigorífico e a parede que separa o refrigerador do congelador, com duas camadas de aço de 0,001m de espessura no interior, acopladas, cada uma, com uma camada de isolante de 0,055m e, no exterior, uma camada de plástico de 0,001m, que contactam com o ar do refrigerador e do congelador.

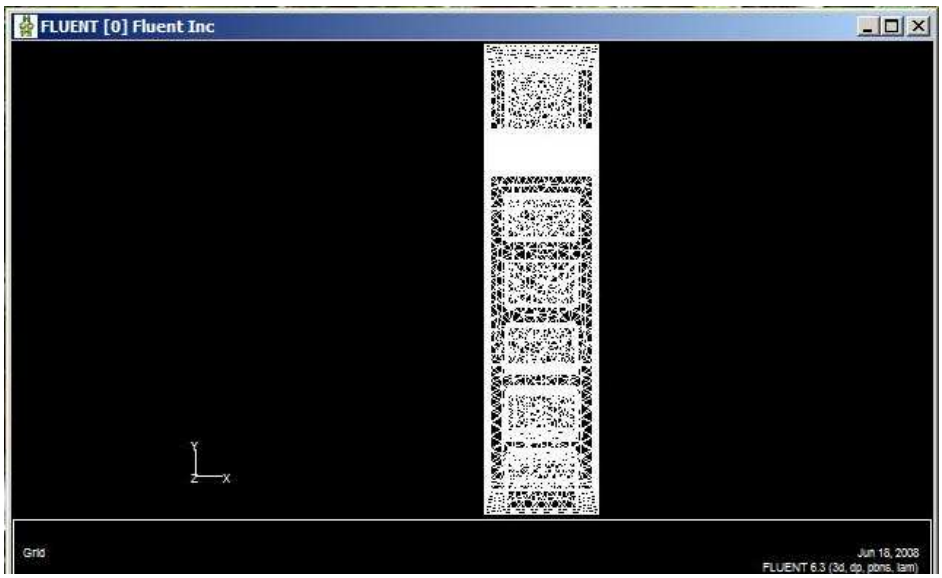


Ilustração 3: Malha da estrutura do frigorífico

Ao ar interior do refrigerador foram subtraídas as localizações indicadas no programa, 4 prateleiras, 1 gaveta e os locais para guardar as garrafas e ovos, na porta. Estas localizações pretendiam indicar o sítio onde os produtos iriam ser guardados, e foram representadas como paralelepípedos de um volume cerca de 10% menor que o da localização real. No caso do ar do congelador, apenas foi representado o paralelepípedo que representa a carga naquele local.

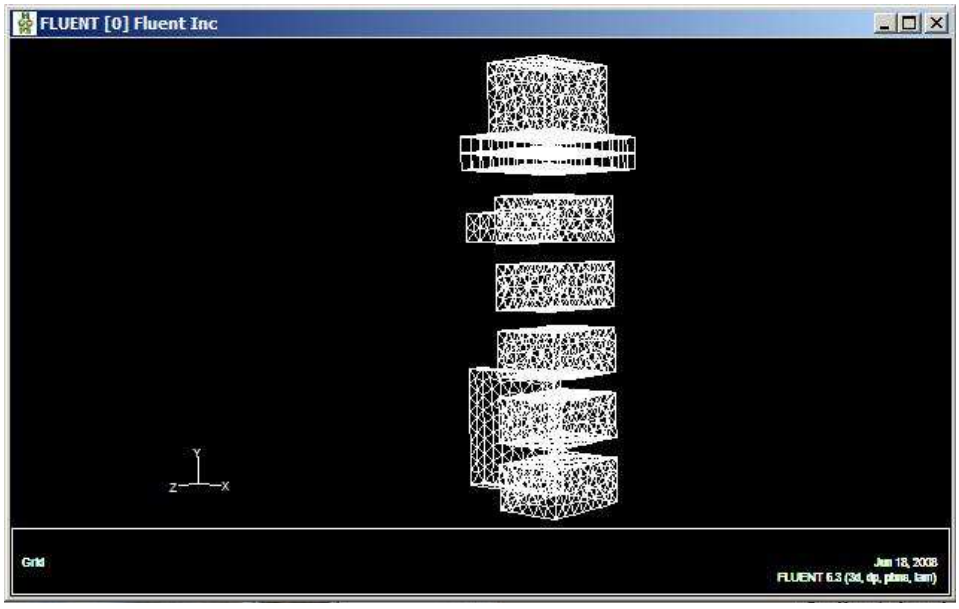


Ilustração 4: Malha das diversas localizações do frigorífico

As malhas usadas para os volumes representados foram de elementos híbridos do tipo T-Grid com 0,05m de espaçamento para o ar interior do refrigerador e congelador e de elementos hexagonais do tipo Cooper para a parede entre o refrigerador e o congelador. Foram utilizados cerca de 8700 elementos para representar o volume do ar do

refrigerador e cerca de 1900 para o congelador. Embora a malha seja relativamente grosseira, esta foi uma necessidade imposta pelo tempo de cálculo do FLUENT para uma malha mais apertada, e é compensada pelo pequeno tempo que cada passo cobre, 2 minutos. A malha foi posteriormente refinada pelo FLUENT.

Foram definidas, no GAMBIT, as zonas de tipo contínuo, fluidos e sólidos e as fronteiras utilizadas pelo FLUENT. As zonas foram definidas como se segue:

Tabela 12: Tipos de fronteiras definidas no GAMBIT

Volume	Tipo de zona
Ar do refrigerador	fluido
Ar do congelador	fluido
Parede fria – evaporador - refrigerador	sólido
Parede fria – evaporador - congelador	sólido
Placa de aço – lado do congelador	parede
Placa de isolamento - congelador	parede
Placa de plástico - congelador	parede
Placa de aço – lado do refrigerador	parede
Placa de isolante - refrigerador	parede
Placa de plástico - refrigerador	parede
1ª prateleira	parede
2ª prateleira	parede
3ª prateleira	parede
4ª prateleira	parede
gavetas	parede
Garrafas – na porta	parede
Ovos – na porta	parede
Carga do congelador	parede
Evaporador - refrigerador	parede
Evaporador - congelador	parede

4. PROGRAMAÇÃO DESENVOLVIDA

Para a criação deste programa de cálculo de cargas térmicas de produtos perecíveis foram utilizados três programas: o MATLAB, o GAMBIT e o FLUENT. Estes programas foram os escolhidos por serem os que melhor se adaptavam às necessidades de cálculo e simulação exigidas para o programa desenvolvido. As versões utilizadas foram, respectivamente, o MATLAB R2007, GAMBIT 2.2.30 e FLUENT 6.3.26, licenciados para uso na Faculdade de Engenharia da Universidade do Porto. O programa desenvolvido foi apelidado de CalCar.

4.1. Programação no FLUENT

Para a simulação da evolução das cargas libertadas pelos produtos perecíveis nas diversas localizações do frigorífico houve a necessidade de conhecer a evolução das temperaturas no interior do mesmo. No programa MATLAB não existe nenhuma ferramenta capaz de realizar a simulação desejada em 3D e, devido a limitações de tempo, não era viável o desenvolvimento destas, pelo que foi seguida a opção de utilizar um programa de CFD externo. O programa escolhido, devido à sua flexibilidade, capacidades de simulação e possibilidade de uso em modo de linhas de comando foi o FLUENT. No programa FLUENT foram importadas as malhas criadas para os diferentes frigoríficos (com ou sem congelador, e para as diferentes capacidades) e as zonas definidas pelo GAMBIT. Foram de seguida definidas as equações que iriam ser utilizadas na simulação.

Foi escolhido o solver para regime transitório e seleccionada a resolução da equação de energia.

O FLUENT resolve a equação de energia na seguinte forma:

$$\frac{\delta}{\delta t}\cdot(\rho\cdot E)+\nabla\cdot\left(\vec{v}\cdot(\rho\cdot E+p)\right)=\nabla\cdot\left(k_{eff}\cdot\nabla T\sum_{j}h_j\cdot\vec{J_J}+\left(\bar{\bar{\tau}}_{eff}\cdot\vec{v}\right)\right)+S_h$$

ρ – massa volúmica (kg/m^3);

E – energia do fluido (J);

h – entalpia do fluido (J/kg);

p – pressão do fluido (Pa);

Onde os três primeiros termos da equação representam a transferência de calor por condução, difusão e dissipação viscosa, k_{eff} é a condutividade efectiva, S_h inclui quaisquer fontes de calor volumétricas definidas pelo utilizador e J_j é o caudal de difusão de espécies.

Foi escolhido o modelo de radiação DO (Discrete Ordinates), uma vez que era o que oferecia os melhores resultados na presença de fontes de calor, como os produtos. A temperatura de Boussinesq escolhida, necessária para o modelo de convecção natural, foi de 278K. Também foi definida a aceleração gravítica de 9,81 m/s^2.

Em seguida criaram-se os materiais sólidos existentes na representação do frigorífico, o aço, a espuma de poliuretano e poliestireno expandido e o plástico. O fluido existente no frigorífico, o ar, vem já pré-definido no FLUENT. As suas propriedades foram definidas como se segue:

Tabela 13: Propriedades dos materiais do frigorífico

Materiais Propriedades	aço	poliuretano	poliestireno	plástico
massa volúmica (kg/m^3)	7860	40	22	1050
calor específico (J/kg.K)	620	1450	1130	1300
condutividade térmica (W/m.K)	45	0,027	0.22	0.52

Materiais	aço	poliuretano	poliestireno	plástico
espessura (m)	0.001	0.055	0.055	0.001

Em regiões sólidas, o FLUENT utiliza a seguinte equação de transporte:

$$\frac{\delta}{\delta t} \cdot (\rho \cdot h) + \nabla \cdot (k \cdot \nabla T) + S_h$$

ρ – densidade (kg/m3);

h – entalpia sensível (J/kg.K);

k – condutividade (W/m.K);

T – temperatura (K);

S_h – fonte de calor volumétrica;

representando o segundo termo da equação, a energia de convecção devido a movimento dos sólidos. Foi, portanto, considerado nulo nesta simulação.

Após todas as anteriores condições e variáveis definidas, foram indicadas as condições de fronteira. Para a simulação inicial, foi considerado que o frigorífico se encontrava em regime permanente, em que todos os produtos que nele se encontravam já não emitiam calor, e se encontravam à temperatura do frigorífico, como indicado nas hipóteses simplificativas. As condições fronteira iniciais foram então assentes como:

Tabela 14: Condições iniciais introduzidas no FLUENT, fronteiras

Volume	Temperatura (K)	Fluxo de calor (W/m^2)

1ª prateleira	278	0
2ª prateleira	278	0
3ª prateleira	278	0
4ª prateleira	278	0
gavetas	278	0
garrafas	278	0
ovos	278	0
carga congelador	255	0
parede refrigerador	278	0
parede congelador	255	0
refrigerador – aço	278	0
refrigerador – isolante	278	0
refrigerador – plástico	278	0
congelador – aço	255	0
congelador – isolante	255	0
congelador - plástico	255	0

Tabela 15: Condições iniciais introduzidas no FLUENT, volumes

Volumes	Participa na radiação	Valores fixos
ar do refrigerador	Sim	Não
ar do congelador	Sim	Não
evaporador no refrigerador	Sim	272K
evaporador no congelador	Sim	250K

As condições de fronteira para os passos seguintes são definidas pelo programa, e vão ser colocadas como fluxos de calor, em W/m^2, nas respectivas localizações. As cargas pela envolvente, por infiltrações e por renovações de ar são adicionadas na parede do refrigerador e parede do congelador, que se tratam, não das paredes dos respectivos locais, mas da sua malha interior, estando em contacto com o ar do refrigerador e do congelador, respectivamente.

Por fim, foi definido o tempo de cada passo como sendo de 120 segundos, com 20 iterações por cada passo.

Com a utilização do FLUENT veio a necessidade de criar alguns ficheiros externos ao programa desenvolvido, como os ficheiros *.jou, que são ficheiros que contêm sequências de ordens a executar no FLUENT em modo batch. Também são criados

ficheiros *.cas e *.dat, ficheiros de definições do caso a estudar e dados originados pela simulação, respectivamente.

4.2. Programação no MATLAB

O software escolhido de como base para o programa desenvolvido foi o MATLAB. Esta escolha deve-se às inúmeras vantagens que este oferece: é um programa de cálculo matricial, pelo que são dispensados ciclos de cálculos em diversas operações do programa desenvolvido; é um programa extremamente versátil, conseguindo criar interfaces para o utilizador, o que permite a criação de programas de cálculo complexos passíveis de serem usados por qualquer utilizador, que, mesmo que não conheça a teoria por detrás do programa, obtém os resultados desejados. Esta versatilidade é também vista na importação de ficheiros exteriores ao MATLAB, com a possibilidade da sua edição e posterior recolocação no seu formato original. Também a capacidade de executar comandos de MS-DOS foi crucial para a ligação entre o MATLAB e o FLUENT. Foi, portanto, a melhor opção, de entre as analisadas, para a criação do programa, sendo em seguida descritas as componentes do mesmo.

4.2.1. GUI's (Graphical User Interface)

A programação no MATLAB iniciou-se, como já foi indicado, pela criação dos GUI's.

O GUI principal define a estrutura e variáveis relativas ao frigorífico: se existe ou não um congelador (por defeito, é escolhida a opção com congelador), o seu tipo de isolamento (por defeito é a espuma de poliuretano), o seu volume (200L por defeito), as temperaturas ambiente, do refrigerador e do congelador (20ºC, 5ºC e -18ºC respectivamente, por defeito). Se é considerada a existência de um congelador no frigorífico, esta localização aparece como opção visível no menu de distribuição dos produtos, se não existe um congelador, esta opção desaparece do menu de distribuição e, mesmo que já tenha sido seleccionada anteriormente (na opção com congelador activa), desaparece o painel congelador do GUI. São também definidos o número diário de vezes que as portas do frigorífico são abertas (10 vezes para a porta do refrigerador e 3 para a do congelador, por defeito) e o tempo médio de abertura das mesmas (15s, por defeito), o estado das membranas das portas do frigorífico (por defeito são consideradas novas), é aberto o acesso aos GUI's das localizações ao aceder ao menu de distribuição dos produtos, é definido o tempo de simulação (24h por defeito) e, após a realização dos cálculos e completa simulação, é permitido também o acesso ao GUI dos gráficos.

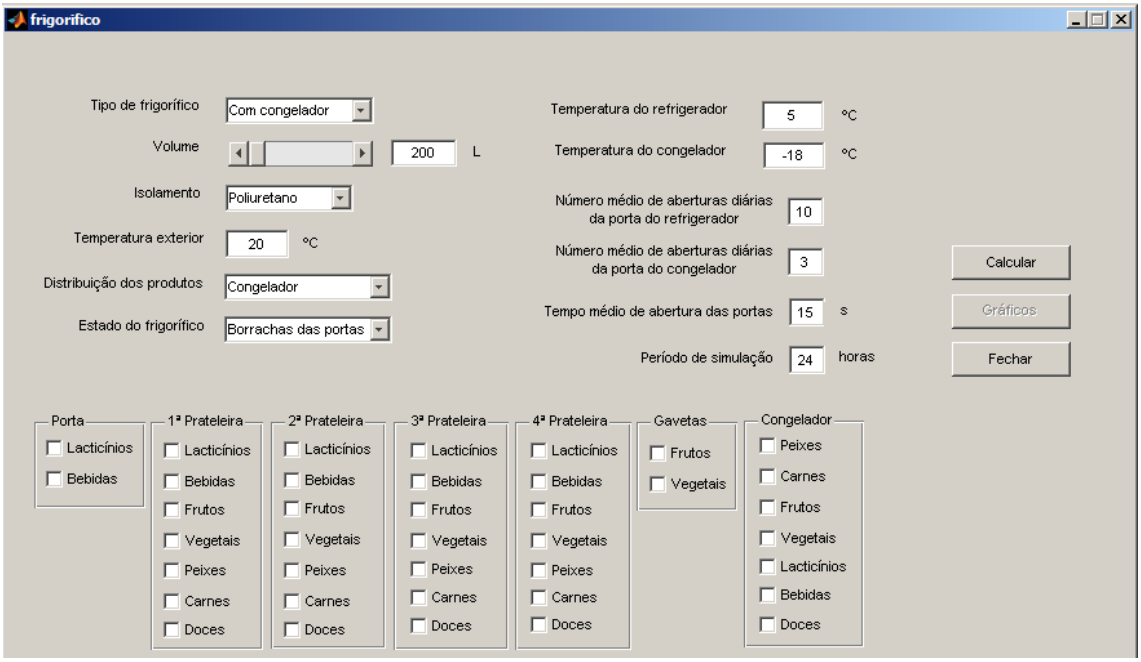


Ilustração 5: GUI principal

Para os GUI's dos produtos foram escolhidas como entradas a quantidade do produto e onde os produtos se podem adquirir ou por unidade ou por peso, também foi colocado um menu para escolher as unidades da quantidade, e também foi criado um menu para escolher a temperatura a que o produto se encontra para entrar, ou existe já, no frigorífico. Foram também colocados dois botões, o botão "OK", que guarda as entradas feitas pelo utilizador nas matrizes criadas respectivas, vê se o volume de produtos seleccionados ultrapassa o volume da localização onde foram colocados e fecha o GUI; e o botão "Cancelar", que descarta estas entradas e fecha o GUI.

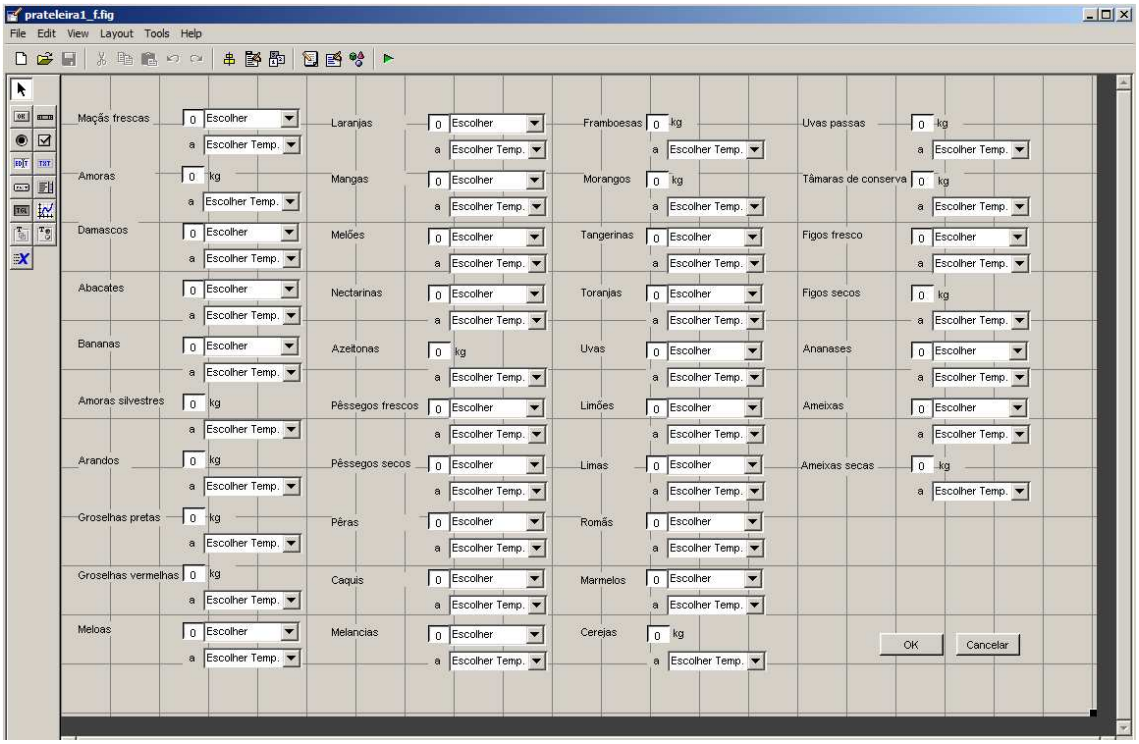


Ilustração 6: GUI frutos em modo de edição

Em cada GUI é seleccionada, primeiro, a quantidade do produto, depois a unidade a que essa quantidade se refere e por fim a temperatura a que se encontra. Esta sequência é obrigatória e, se o utilizador não seleccionar as unidades da quantidade ou a temperatura do produto, é exibida uma mensagem de erro que indica a ordem a que se devem colocar as entradas. Se, após escolher a quantidade, unidade e temperatura de um produto o utilizador decidir mudar a quantidade, os menus de unidades e temperatura voltam ao valor inicial, para escolher as novas condições.

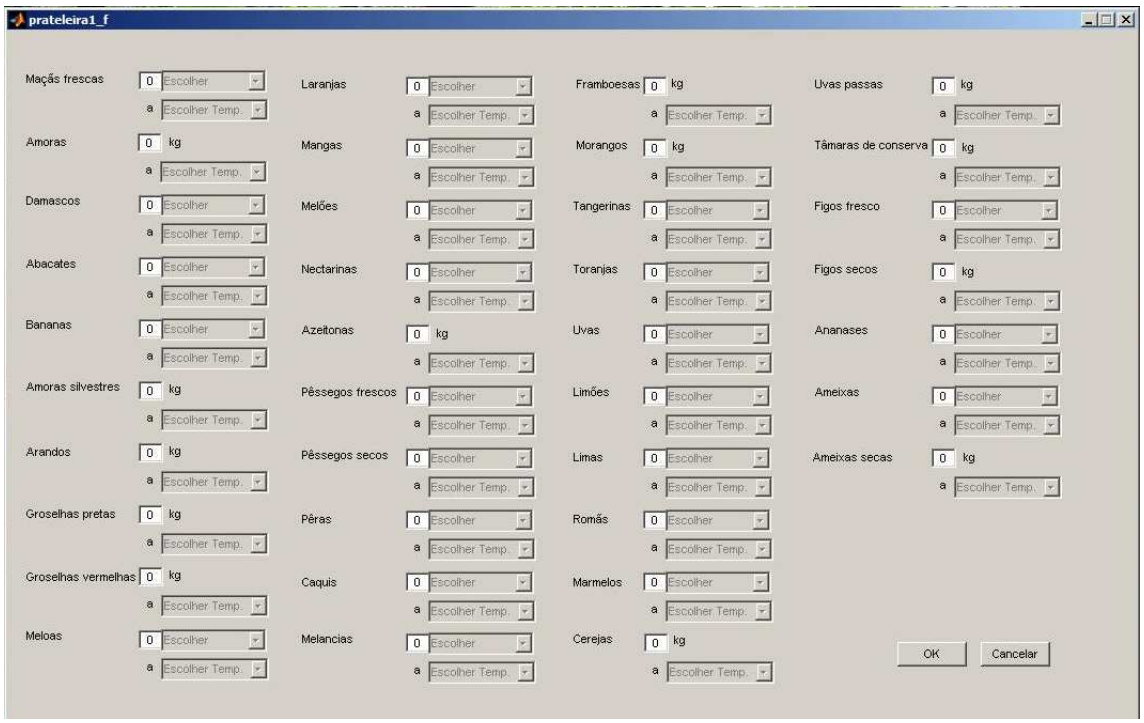


Ilustração 7: GUI frutos inicializado

Quando os GUI's são chamados pelo GUI principal, apenas as entradas de quantidades dos produtos são possíveis de alterar. Isto para que a ordem de entradas de valores do utilizador seja mais simples, sendo também mais fácil a visualização do que já foi escolhido.

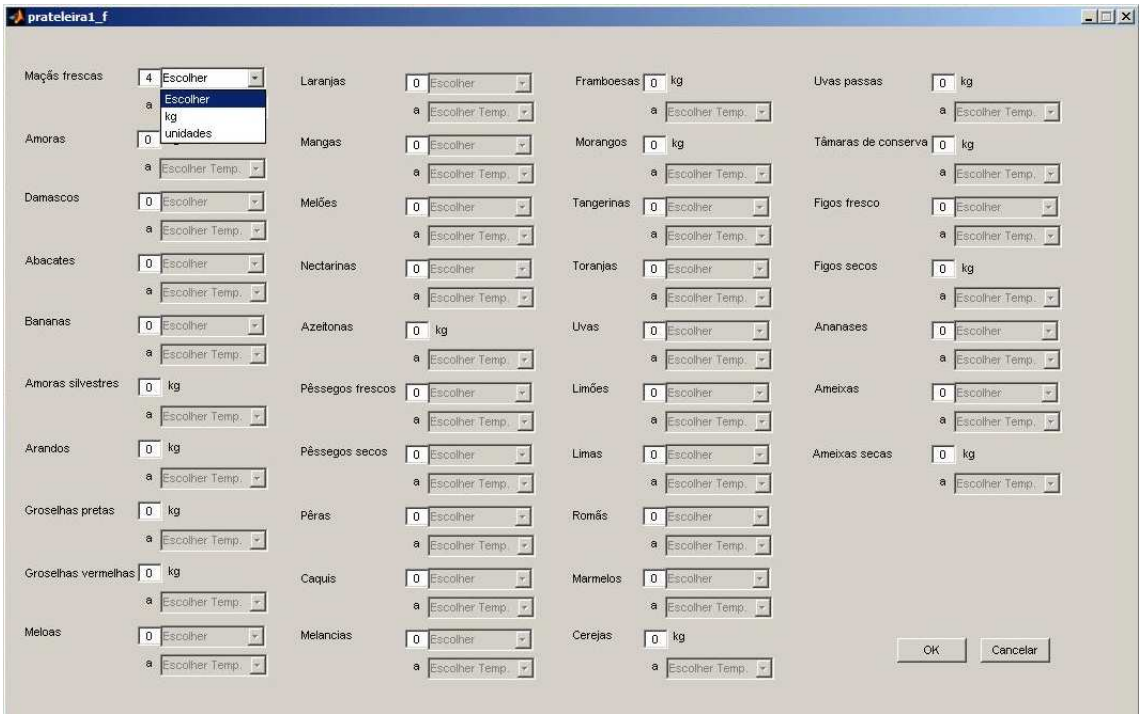


Ilustração 8: GUI frutos após selecção da quantidade

Esta ordem de entradas deve-se ao facto de os cálculos de conversão (de L para kg), de calor emitido por diferenças de temperaturas e do calor emitido por respiração e por transpiração serem feitos aquando da escolha da temperatura, quando já existem todos os dados necessários. A colocação destes mesmos cálculos nas outras entradas era também possível, mas acarretaria um código três vezes maior, com os consequentes maiores tempos de processamento e utilização de memória.

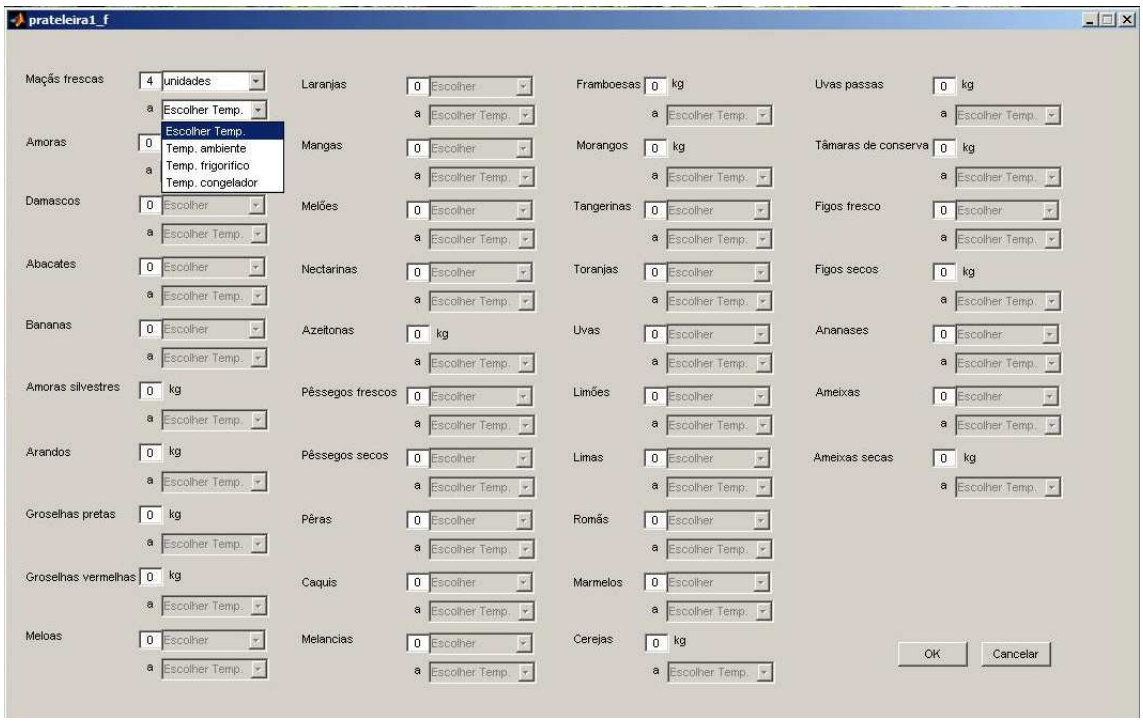


Ilustração 9: GUI frutos após a selecção da quantidade e unidades

Para visualizar os resultados foi criado o GUI gráficos, que permite a obtenção dos gráficos já anteriormente mencionados.

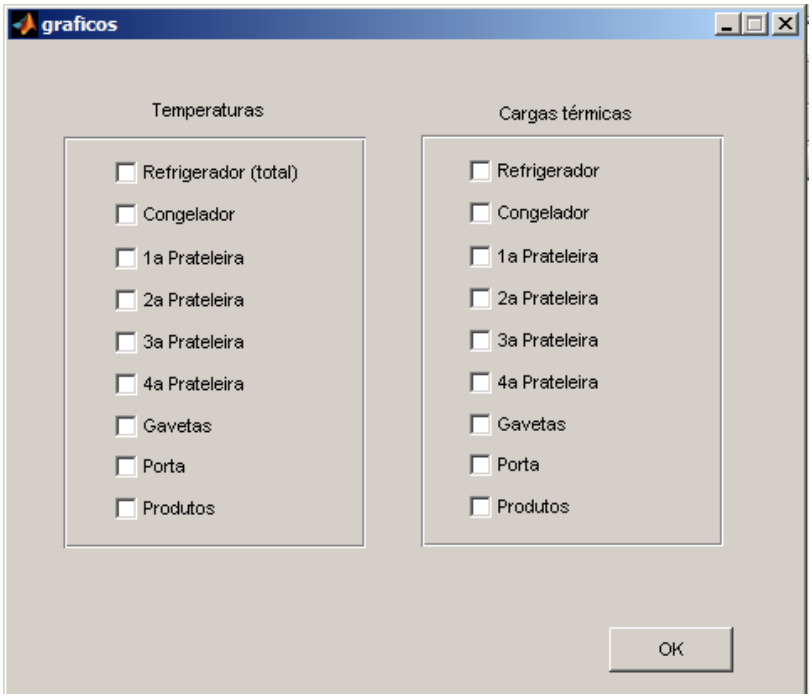


Ilustração 10: GUI de escolha de gráficos

Após a construção dos GUI's foram criadas as matrizes necessárias para o armazenamento da informação inicial e inalterável sobre os elementos constituintes dos produtos e as matrizes auxiliares, que são iniciadas sempre vazias, utilizadas para guardar a informação inserida pelo utilizador e os resultados dos cálculos. Serão brevemente descritas em seguida.

4.2.2. Matrizes de informação

Sendo o MATLAB um programa, na sua base, de cálculo matricial, os diversos valores das propriedades dos produtos perecíveis, assim como as entradas de dados pelo utilizador são guardados em matrizes, que são descritas em seguida.

- Matriz propriedades do ar: contém os calores específicos, a entalpia e valores de Prandtl, para diversas temperaturas.

- Matriz pressão de saturação: contém os valores da pressão de saturação e entalpia de vapor saturado para temperaturas entre os -30ºC e os 50ºC.

- Matrizes calor_*: estas matrizes contêm os valores totais das potências libertadas pelos produtos, em cada localização do frigorífico. Para a 1ª prateleira é utilizada a matriz calor_1p, calor_2p para a 2ª prateleira, calor_3p para a 3º prateleira, calor_4p para a 4ª prateleira, calor_g para a gaveta, calor_p para a porta e calor_c para o congelador. Os valores contidos nestas matrizes serão utilizados para a criação dos gráficos de resultados.

- Matriz propriedades das bebidas: contém as percentagens de constituintes para diversas bebidas, assim como o ponto inicial de congelamento, os calores específicos acima e abaixo do ponto de congelamento e o calor latente das bebidas. Como

propriedades definidas tem as dimensões características e rácios dimensionais utilizados no cálculo dos tempos de refrigeração e congelamento, o peso individual das bebidas (a sua massa, em kg) e guarda a quantidade de produtos definidas pelo utilizador (convertida para massa), a temperatura a que se encontram e a sua localização. Também guarda o volume ocupado pelos produtos em cada localização.

- Matriz propriedades das carnes: igual á matriz propriedades das bebidas, com diferentes tipos de carnes.

- Matriz propriedades dos doces: igual á matriz propriedades das bebidas, com diferentes tipos de doces.

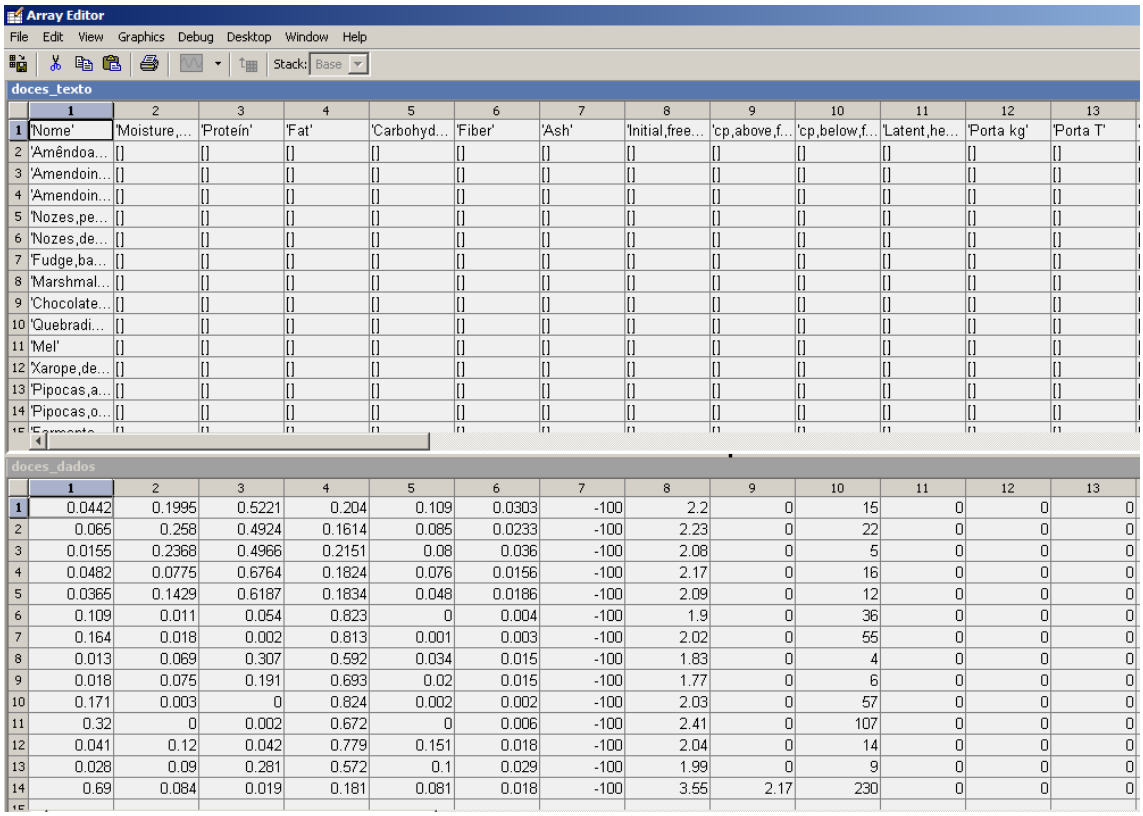


Ilustração 11: Vista parcial da matriz doces

- Matriz propriedades dos frutos: igual á matriz propriedades das bebidas, com diferentes tipos de frutos.

- Matriz propriedades dos lacticínios: igual á matriz propriedades das bebidas, com diferentes tipos de lacticínios.

- Matriz propriedades dos peixes: igual á matriz propriedades das bebidas, com diferentes tipos de peixes.

- Matriz propriedades dos vegetais: igual á matriz propriedades das bebidas, com diferentes tipos de vegetais.

- Matrizes r_1ap_*: guardam os valores das potências caloríficas libertadas pelos produtos na primeira prateleira do refrigerador, * é b na matriz das bebidas r_1ap_b, c na matriz das carnes r_1ap_c, d na matriz dos doces r_1ap_d, f na matriz dos frutos

r_1ap_f, l na matriz dos lacticínios r_1ap_l, p na matriz dos peixes r_1ap_p, v na matriz dos vegetais r_1ap_v, f_resp para matriz que guarda a potência calorífica libertada pela respiração dos produtos r_1ap_f_resp e v_resp para a matriz que guarda a potência calorífica libertada pela respiração dos vegetais r_1ap_v_resp. Também são guardados os valores das temperaturas de cada produto nas matrizes r_1ap_*_temp. Estes valores são guardados em cada passo de simulação do FLUENT.

- Matrizes r_2ap_*: idênticas às matrizes r_1ap_*, mas guardam os valores das potências caloríficas libertadas pelos produtos na segunda prateleira.

- Matrizes r_3ap_*: idênticas às matrizes r_1ap_*, mas para a terceira prateleira.

- Matrizes r_4ap_*: idênticas às matrizes r_1ap_*, mas para a quarta prateleira.

- Matrizes r_g_*: idênticas às matrizes r_1ap_*, mas apenas para guardar as potências caloríficas libertadas pelos frutos, vegetais, e respiração dos mesmos, na gaveta.

- Matrizes r_p_*. Idênticas às matrizes r_1ap_*, mas apenas para guardar as potências caloríficas libertadas pelas bebidas e lacticínios, na porta.

- Matrizes c_*: idênticas às matrizes r_1ap_*, mas para o congelador.

- Matriz r_envolv: guarda os valores da potência calorífica que atravessa da envolvente para o frigorífico, em cada passo de simulação. Estão incluídos nesta matriz os valores da potência que entra no frigorífico devida às renovações de ar, aquando da abertura de portas, uma vez que este valor foi calculado diariamente e a sua contribuição para cada passo da simulação é a parcela correspondente a dois minutos do valor calculado.

- Matriz r_infilt: guarda os valores da potência calorífica devida às infiltrações em cada passo da simulação.

- Matriz temp_frig: esta matriz guarda as temperaturas de todos os locais do frigorífico, 1ª prateleira, 2ªa prateleira, 3ª prateleira, 4ª prateleira, gaveta, porta, congelador e ar do refrigerador, em cada passo.

- Matriz valor ómega: esta matriz foi criada pela incapacidade do MATLAB de resolver a equação transcendental necessária para o cálculo dos tempos de refrigeração. Foram calculados vários resultados dessa equação e os respectivos números de Biot que os originaram, e guardados nesta matriz.

Após a definição das matrizes necessárias, foram criadas as funções e procedimentos que realizam os cálculos desejados.

4.2.3. Funções e procedimentos

Para a automatização dos cálculos foram criadas diversas funções que, com entradas específicas, definidas no texto de ajuda de cada função, retornam o resultado ou vários resultados dos cálculos desejados. Também para automatizar sequências de instruções e utilização das funções foram criados procedimentos. Ambos são descritos em seguida, apresentados por ordem alfabética. O código destas funções e procedimentos encontra-se no anexo A.

- Função alfas: esta função determina as temperaturas das paredes exteriores e interiores do refrigerador ou do congelador, em ºC, para servirem como entrada às funções de cálculo das potências introduzidas pela envolvente. Tem como entradas as temperaturas ambiente, do refrigerador e do congelador, em ºC; a altura, largura e profundidade, em m, das paredes; a indicação de se tratar do refrigerador ou do congelador e as indicações de se tratar de paredes laterais, de topo ou de fundo. Calcula os números adimensionais de Grashof, Raleygh, Prandtl e Nusselt para determinar o coeficiente de transferência de calor por convecção necessário ao cálculo das resistências térmicas de convecção das paredes, através das quais se determinam as temperaturas das mesmas.

- Função calor_paredes_c: esta função determina a potência térmica, em W, que atravessa as paredes do congelador. Considera como entradas as temperaturas ambiente, do congelador e as temperaturas das paredes laterais, de topo e de fundo, em ºC, a altura, largura e profundidade, em m, das paredes do congelador. Calcula as resistências por convecção, condução e radiação das paredes, para determinar a resistência térmica total que permitirá conhecer as potências introduzidas pela envolvente.

- Função calor_paredes_r: esta função determina a potência térmica, em W, que atravessa as paredes do refrigerador. Tem como entradas as temperaturas ambiente, do refrigerador, e do congelador, em ºC, a indicação de se tratar do refrigerador ou do congelador e as dimensões das paredes e respectivas camadas, em m.

- Funções calor_prod_*: determinam a potência térmica, em W, libertada pelos produtos, tratando-se de bebidas em calor_prod_b, carnes em calor_prod_c, lacticínios em calor_prod_l e peixes em calor_prod_p, desde a sua temperatura inicial até à sua temperatura final, devida à diferença de temperaturas entre o meio envolvente e o produto; Tem como entradas o tipo e a linha do produto na matriz das suas propriedades, a sua temperatura, em ºC, a sua massa, em kg, e o local onde vai ser armazenado. Calcula as propriedades dos constituintes do produto, o seu calor específico, a potência térmica por ele libertado e a nova temperatura devida a essa libertação de energia.

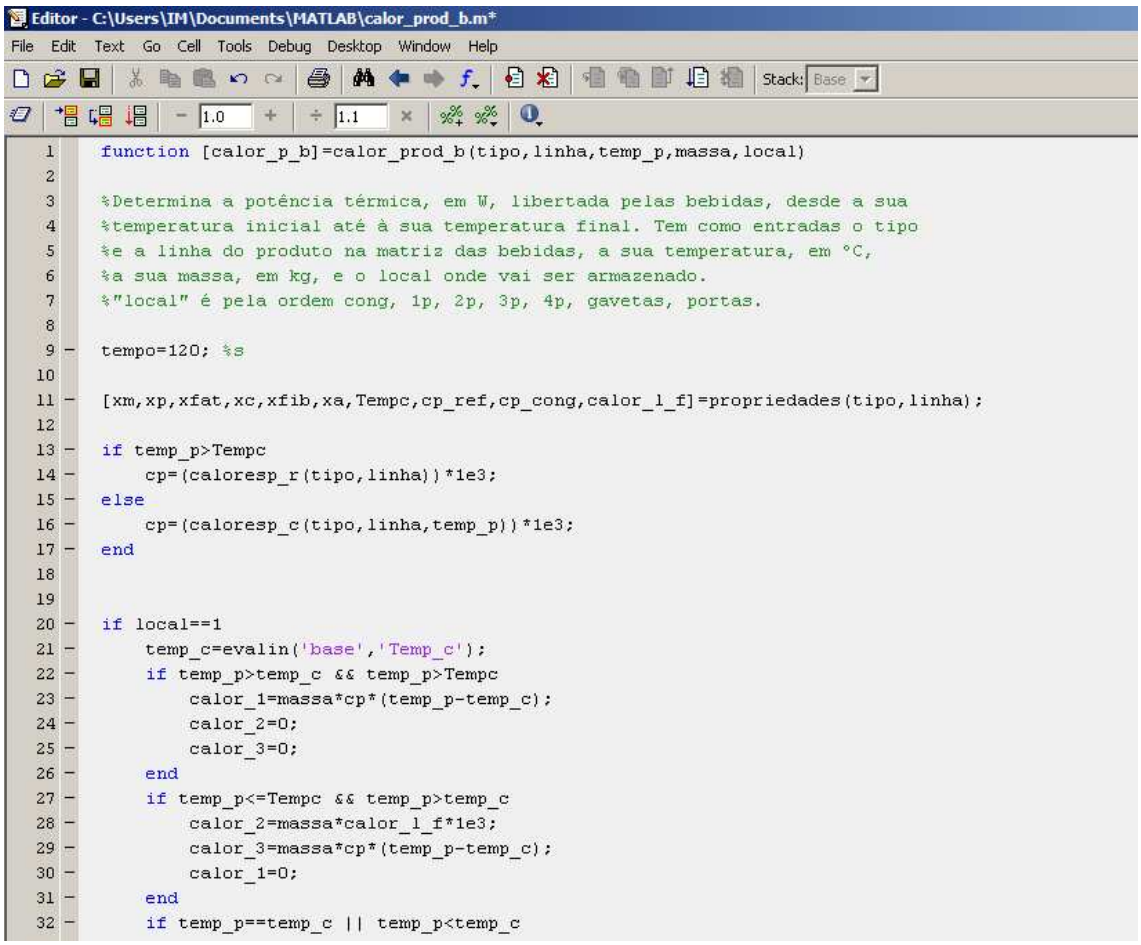


Ilustração 12: Vista parcial da função calor_prod_b

- Funções calor_prod_*: iguais às funções anteriores, mas para os frutos, em calor_prod_f e vegetais, em calor_prod_v. Estas funções têm o cálculo adicional das potências libertadas por respiração e por transpiração dos produtos dos quais se conhecem estas informações.

- Função caloresp_c: calcula o calor específico aparente de um produto, em kJ/kg.K, abaixo da sua temperatura de congelamento. Tem como entradas o tipo e linha do produto na sua respectiva matriz de propriedades e a temperatura, em ºC, abaixo da sua temperatura inicial de congelamento, a que se encontra.

- Função caloresp_r: calcula o calor específico de um produto, em kJ/kg.K, acima da sua temperatura de congelamento. Tem como entrada o tipo e a linha do produto na respectiva matriz de propriedades.

- Procedimento checkfiles: este procedimento faz uma procura dos ficheiros “temp1ap”, “temp2ap”, “temp3ap”, “temp4ap”, “tempg”, “tempgarr”, “temparrefr” e “temparcong”, que são, respectivamente, as temperaturas da 1ª, 2ª, 3ª e 4ª prateleiras, temperaturas da gaveta, garrafas, ar do refrigerador e ar do congelador. Estes ficheiros são exportados pelo FLUENT e indicam os valores nodais de temperatura para cada localização do frigorífico. Após detectar a existência de todos estes ficheiros, o procedimento importa-os, convertendo-os em matrizes que o MATLAB consegue interpretar e faz a média dos

seus valores, em K, transformando depois as temperaturas em ºC e introduzindo esse novo valor no ambiente de trabalho. Após a leitura e conversão das temperaturas, apaga os ficheiros que as originaram, para poderem ser gravados novamente no passo seguinte.

- Função condtermica: a função determina a condutibilidade térmica dos produtos, em W/m.K à temperatura a que se encontram. Tem como entradas o tipo e linha do produto na respectiva matriz de propriedades e a temperatura a que o produto se encontra, em ºC.

- Função densidade: calcula a massa volúmica, kg/m^3, dos produtos, assim como dos seus componentes, para a temperatura a que os produtos se encontram. Tem como entradas o tipo e a linha do produto na respectiva matriz de propriedades e a sua temperatura, em ºC.

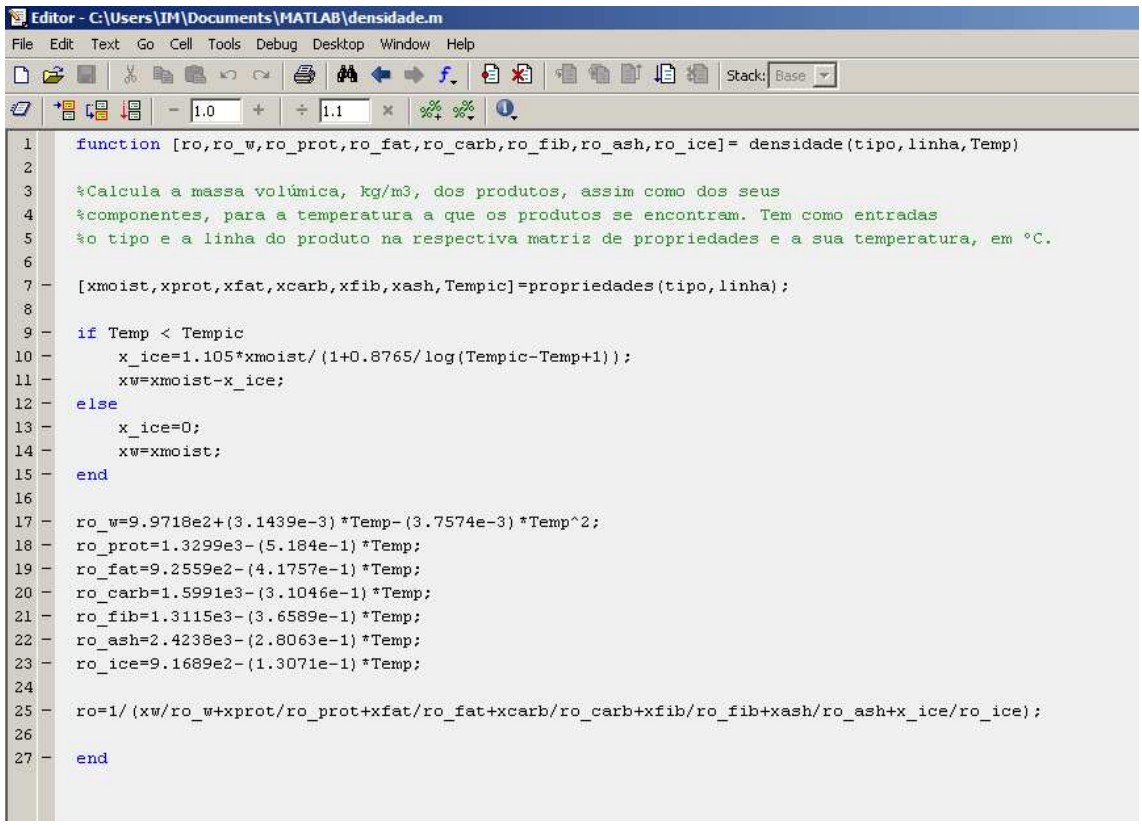


Ilustração 13: Vista parcial da função densidade

- Função entalpia_c: determina a entalpia do produto, em kJ/kg, abaixo da sua temperatura de congelamento. Tem como entradas o tipo e a linha do produto na respectiva matriz de propriedades e a sua temperatura, em ºC, abaixo do valor de congelamento.

- Função entalpia_r: determina a entalpia dos produtos, em kJ/kg, acima da sua temperatura de congelamento. Tem como entradas o tipo e a linha do produto na respectiva matriz de propriedades e as temperaturas do produto e de congelamento, em ºC.

- Funções import_file e importfile_sim: estas funções importam os ficheiros que contêm as ordens a executar no FLUENT e convertem-nos em matrizes que o MATLAB consegue editar.

- Função infiltr: calcula a taxa de ar infiltrado (s^{-1}) e a respectiva potência, em W, que entra no frigorífico através das infiltrações que as membranas das portas deixam passar para o interior, num determinado período de tempo. Tem como entradas o período de tempo, em minutos, para o qual se quer calcular a potência infiltrada, a indicação de se tratar de um frigorífico novo ou velho, com as respectivas membranas a serem consideradas também, respectivamente, novas ou velhas; a indicação de o cálculo estar a ser efectuado para o congelador ou para o refrigerador, o volume de ar existente no local onde se querem calcular as infiltrações e as temperaturas ambiente, do refrigerador e do congelador, em ºC.

- Função propriedad_ar: esta função calcula as propriedades do ar à sua temperatura. As propriedades calculadas são a massa volúmica, em kg/m^3, a entalpia, em kJ/kg, o calor específico, em kJ/kg.K, a condutibilidade térmica, em W/m.K, a viscosidade dinâmica, em $N.s/m^2$, a viscosidade cinemática, em m^2/s, o coeficiente β em K^{-1} e o número de Prandtl. A função tem como entrada a temperatura do ar, em ºC. Os cálculos são efectuados através da matriz propr_ar, onde o MATLAB faz uma aproximação polinomial aos valores existentes de temperatura e propriedade a calcular e depois calcula o valor dessa propriedade à temperatura do ar.

- Função propriedades: esta função devolve as propriedades dos produtos, tendo como entrada o tipo e a linha do produto na respectiva matriz de propriedades, onde são apenas lidos os seus valores e atribuídos às designações correctas.

- Função q_transp: calcula a potência, em W, libertada pela transpiração dos produtos e vegetais. Tem como entradas o tipo e linha do produto na respectiva matriz de propriedades, as temperaturas do produto e do ar envolvente, em ºC e o local onde o produto se encontra ou vai ser guardado. Esta função aproxima os valores da matriz pressão_sat através de uma aproximação polinomial e calcula o valor da pressão, em Pa e da entalpia de vapor saturado, em kJ/kg.K, valores necessários à determinação do calor libertado por transpiração.

- Procedimento Q_trocado: este procedimento chama as funções calor_prod_* para cada passo da simulação, soma todos os valores de potência produzidos pelos diferentes produtos em cada localização do frigorífico e grava-os nas matrizes calor_*. Também, neste procedimento, são importados os ficheiros que contêm as ordens a executar no FLUENT, onde, nas respectivas posições, são colocados os novos valores de potência libertada calculada.

- Função renov_ar: calcula o valor da potência que entra no frigorífico, em W, devida à abertura das portas. Tem como entradas a indicação de se tratar do cálculo para o congelador ou para o refrigerador, a área da porta do local respectivo, em m^2, e a altura da mesma porta, em m.

- Função tempo_c: determina o tempo de congelamento de um produto, em s, desde a sua temperatura inicial até à temperatura da sua envolvente. Tem como entradas o tipo e a linha do produto na respectiva matriz de propriedades e as temperaturas do produto e

do congelador, em ºC. O cálculo efectua-se para uma temperatura final do centro térmico do produto igual à da sua envolvente. Para a obtenção do tempo de descongelamento, foi considerado um valor quatro vezes maior que o dado por esta função, como indicado na sebenta de Refrigeração,2006, do Professor Clito Afonso [(4)].

- Função tempo_r: determina o tempo de refrigeração de um produto, em s, desde a sua temperatura inicial até à sua temperatura final, considerada como 0.5ºC acima da temperatura da sua envolvente, por impossibilidade do cálculo do tempo de refrigeração, com a formulação usada, estando o centro térmico do produto e a sua envolvente à mesma temperatura. Tem como entradas o tipo e a linha do produto na respectiva matriz de propriedades e as temperaturas do produto e de refrigeração, em ºC. Descrevem-se, em seguida, as variáveis utilizadas no programa.

Para serem determinados os coeficientes de transferência de calor à superfície dos produtos foram considerados os valores dados na ASHRAE Refrigeration;2002(1). Foram utilizados os valores de carne de bovino para as carnes, 10 W/m^2.K, uma vez que a temperatura tabelada era de -19,5ºC e estes produtos são habitualmente guardados no congelador, pelo que era a correlação mais próxima. Para os frutos foi utilizado o valor de 23,8 W/m^2.K, o valor tabelado para os figos, devido à sua dimensão ser bastante comum nos produtos frutícolas, e a temperatura tabelada ser de 4ºC, uma temperatura próxima da utilizada de refrigeração. Para os vegetais foi utilizado o valor tabelado para batatas, também por ter uma temperatura próxima da de refrigeração, 4,4ºC, neste caso. Para os doces, uma vez que não foi encontrada nenhuma correlação ou valores indicativos do coeficiente de transferência, foi utilizado o valor dos frutos. Para os lacticínios e para as bebidas, uma vez que para as bebidas também não existiam nem valores nem correlações para o seu cálculo, foi utilizada a correlação seguinte:

$$Nu = 0.754 \cdot Gr^{0.264}$$

Nu – número de Nusselt;

Gr – número de Grashoff;

com,

$$Gr = \frac{\beta \cdot g \cdot \Delta T \cdot L^3}{\nu^2}$$

β – coeficiente de dilatação térmica (K^{-1});

g – aceleração da gravidade (g= 9.8 m/s^2);

ΔT – diferença de temperaturas entre o produto e o ar à sua volta, considerada de 15ºC, nesta função;

L – comprimento característico (m);

υ – viscosidade cinemática (m^2/s);

e,

$$h = \frac{Nu \cdot \lambda}{L}$$

h – coeficiente de convecção à superfície (W/m2.K);

λ – condutibilidade térmica (W/m.K).

O resultado obtido, para propriedades do ar a 5ºC foi de 8 W/m2.K para o leite, com dimensão característica de 0,195 m, um pacote de leite, e de 7,7 W/m2.K para as bebidas, com uma dimensão característica de 0,24 m, um pacote de sumo.

Para os peixes, a correlação utilizada foi a seguinte:

$$Nu = 4.5 \cdot Re^{0.28}$$

Re – número de Reynolds;

e

$$h = \frac{Nu \cdot \lambda}{L}$$

O resultado, para uma dimensão característica de 0,28 m foi de 34,3 W/m2.K.

4.2.4. Variáveis criadas no espaço de trabalho

Foram criadas diversas variáveis, com diversas utilizações, entre as quais serem usadas como armazenadoras temporárias de valores ou terem a função de contadores. Segue-se uma breve descrição das mesmas.

- Temp: esta variável guarda o valor da temperatura ambiente, definida no GUI principal.

- Temp_1ap: guarda o valor da temperatura do ar na 1ª prateleira, em ºC, sendo este valor actualizado para cada passo de simulação. O seu valor inicial é o definido no GUI principal como sendo a temperatura do refrigerador, uma vez que foi considerado o instante inicial como sendo permanente. A actualização faz-se com os valores exportados do FLUENT para esta localização.

- Temp_2ap: como na variável anterior, guarda o valor da temperatura do ar, em ºC, mas na 2ª prateleira, sendo este valor actualizado para cada passo de simulação. O valor inicial e as actualizações da temperatura processam-se como na variável Temp_1ap.

- Temp_3ap: guarda o valor da temperatura do ar na 3ª prateleira, em ºC, sendo este valor actualizado para cada passo de simulação. O valor inicial e as actualizações da temperatura processam-se como na variável Temp_1ap.

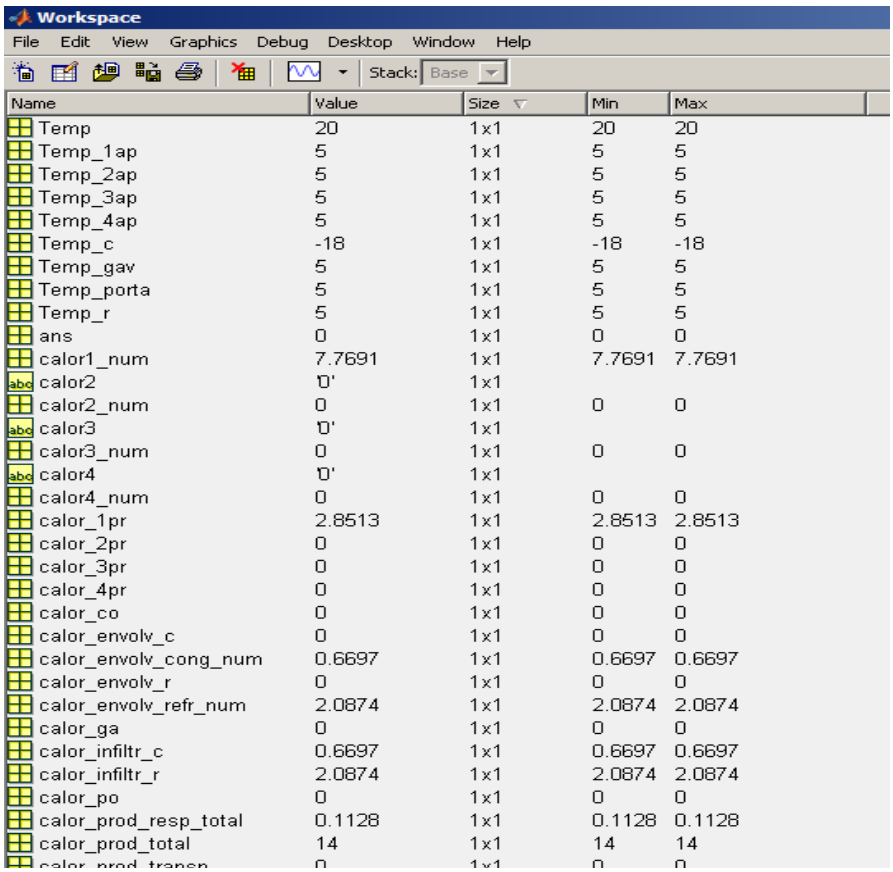
- Temp_4ap: guarda o valor da temperatura do ar na 4ª prateleira, em ºC, sendo este valor actualizado para cada passo de simulação. O valor inicial e as actualizações da temperatura processam-se como na variável Temp_1ap.

- Temp_gav: guarda o valor da temperatura do ar na gaveta, em ºC, sendo este valor actualizado para cada passo de simulação. O valor inicial e as actualizações da temperatura processam-se como na variável Temp_1ap.

- Temp_porta: guarda o valor da temperatura do ar na porta, em ºC, sendo este valor actualizado para cada passo de simulação. O valor inicial e as actualizações da temperatura processam-se como na variável Temp_1ap.

- Temp_r: guarda o valor da temperatura do ar no refrigerador, em ºC, sendo o valor , sendo este valor actualizado para cada passo de simulação. O valor inicial é o definido pelo utilizador no GUI principal e a actualização da temperatura processam-se como na variável Temp_1ap.

- Temp_c: guarda o valor da temperatura do ar no congelador, em ºC, sendo este valor actualizado para cada passo de simulação. O valor inicial é o definido pelo utilizador no GUI principal e a actualização da temperatura processa-se como na variável Temp_1ap.



Name	Value	Size	Min	Max
Temp	20	1x1	20	20
Temp_1ap	5	1x1	5	5
Temp_2ap	5	1x1	5	5
Temp_3ap	5	1x1	5	5
Temp_4ap	5	1x1	5	5
Temp_c	-18	1x1	-18	-18
Temp_gav	5	1x1	5	5
Temp_porta	5	1x1	5	5
Temp_r	5	1x1	5	5
ans	0	1x1	0	0
calor1_num	7.7691	1x1	7.7691	7.7691
calor2	'0'	1x1		
calor2_num	0	1x1	0	0
calor3	'0'	1x1		
calor3_num	0	1x1	0	0
calor4	'0'	1x1		
calor4_num	0	1x1	0	0
calor_1pr	2.8513	1x1	2.8513	2.8513
calor_2pr	0	1x1	0	0
calor_3pr	0	1x1	0	0
calor_4pr	0	1x1	0	0
calor_co	0	1x1	0	0
calor_envolv_c	0	1x1	0	0
calor_envolv_cong_num	0.6697	1x1	0.6697	0.6697
calor_envolv_r	0	1x1	0	0
calor_envolv_refr_num	2.0874	1x1	2.0874	2.0874
calor_ga	0	1x1	0	0
calor_infiltr_c	0.6697	1x1	0.6697	0.6697
calor_infiltr_r	2.0874	1x1	2.0874	2.0874
calor_po	0	1x1	0	0
calor_prod_resp_total	0.1128	1x1	0.1128	0.1128
calor_prod_total	14	1x1	14	14
calor_prod_transp	0	1x1	0	0

Ilustração 14: Variáveis no ambiente de trabalho do MATLAB

- n_vezes_abert: guarda o número de vezes que a porta do refrigerador é aberta diariamente. Este valor é introduzido pelo utilizador no GUI principal.

- n_vezes_abert_cong: guarda o número de vezes que a porta do congelador é aberta diariamente, valor introduzido pelo utilizador no GUI principal.

- novo_velho: pode tomar os valores 1 ou 2. Se o seu valor for 1, indica que o utilizador especificou que o frigorífico era novo, e assim são consideradas as membranas das suas portas. Se o valor for 2, o utilizador especificou que o frigorífico era velho, e também o são consideradas as membranas das portas.

- período_simul: guarda o valor, em horas, do tempo de simulação do programa introduzido pelo utilizador no GUI principal. Este valor é multiplicado por 30, para reflectir o número de simulações de 2 minutos que o programa irá correr.

- tempo_medio_abert: guarda o valor, em segundos, do tempo médio de abertura das portas do frigorífico, valor introduzido pelo utilizador no GUI principal.

- tipo_isol: pode tomar os valores 1 ou 2. Se o valor for 1, indica que o isolamento do frigorífico seleccionado pelo utilizador no GUI principal é de espuma de poliuretano. Se o valor for 2, o isolamento escolhido é de poliestireno.

- vol_preenchimento_*: este conjunto de variáveis guarda os volumes dos produtos colocados em cada localização, vol_preenchimento_1ap na 1ª prateleira, vol_preenchimento_2ap na 2ª prateleira, vol_preenchimento_3ap na 3ª prateleira,

vol_preenchimento_4ap na 4ª prateleira, vol_preenchimento_c no congelador, vol_preenchimento_g na gaveta e vol_preenchimento_p na porta. Os seus valores são a soma dos volumes dos produtos, actualizados de cada vez que se colocam mais produtos, de qualquer tipo, na localização à qual cada variável está atribuída.

- volume: guarda o volume do frigorífico, em L, escolhido pelo utilizador no GUI principal.

- i: é a variável que conta e indica a todas as funções e procedimentos em que passo da simulação o programa se encontra, para que os resultados obtidos sejam colocados nas colunas correspondentes das matrizes de resultados.

- k: é uma variável do tipo texto que transforma o valor da variável i em texto, para ser colocada no ficheiro de ordens a executar no FLUENT.

- calor*_num: estas variáveis guardam o valor da potência libertada por todos os produtos em cada localização do frigorífico dividida pela área dessa localização, ou seja, em W/m2, para introduzir no FLUENT. As variáveis são calor1 para a 1ª prateleira, calor2 para a 2ª prateleira, calor3 para a 3ª prateleira, calor4 para a 4ª prateleira, calorc para o congelador, calorgav para a gaveta e calorp para a porta.

- calor*: este conjunto de variáveis transforma o valor das respectivas variáveis anteriores, calor*_num em texto, para ser introduzido no ficheiro que contém as ordens a executar no FLUENT. São variáveis, portanto, do tipo texto.

- fid: é uma variável definida pelo MATLAB, que guarda a identificação do ficheiro de ordens a executar no FLUENT que está a ser alterado pelo MATLAB.

5. ANÁLISE DE RESULTADOS

Para analisar os resultados obtidos com o programa foi simulado um frigorífico, sendo consideradas a temperatura ambiente de 20ºC, a temperatura interior do refrigerador de 5ºC e a temperatura interior do congelador de -18ºC, no passo inicial. O volume considerado foi de 200L, sendo o frigorífico novo (logo, as membranas de protecção das portas foram também assim consideradas), com isolamento de espuma de poliuretano e com cerca de 10 aberturas diárias da porta do refrigerador e 3 do congelador, ambas com uma média de 15 segundos de duração. O frigorífico foi carregado com a seguinte distribuição de produtos:

- 1ª prateleira: 1 kg maçãs à temperatura ambiente.

- 2ª prateleira: vazia;

- 3ª prateleira: 1 melancia à temperatura ambiente;

- 4ª prateleira: 1 kg de pescada à temperatura do congelador;

- gavetas: 2 alfaces à temperatura ambiente;

- porta: 1 L de leite à temperatura ambiente e 12 ovos.

A temperatura ambiente considerada foi de 20ºC, a temperatura de refrigeração de 5ºC e a de congelamento de -18ºC. A simulação apresentada é da primeira hora de arrefecimento dos produtos, onde os fenómenos de arrefecimento por convecção e radiação e de respiração e transpiração dos produtos são mais dinâmicos. Foram consideradas as infiltrações e cargas introduzidas pela envolvente neste período de tempo. Como resultados, foram pedidas ao programa as cargas devidas à potência libertada em cada localização, assim como a variação da temperatura nestas e a variação de temperatura e potências libertadas pelos produtos, neste tempo. As potências devidas a infiltrações, envolvente e renovações de ar são também dados de saída. Por fim, é pedida a representação do ar interior do refrigerador no fim da simulação.

São apresentados, em seguida, os gráficos das variações de temperatura e cargas térmicas introduzidas pelos produtos perecíveis, em cada localização.

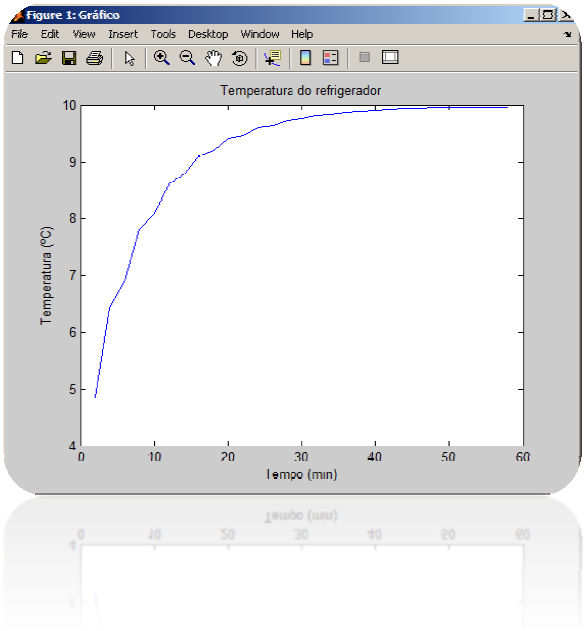


Ilustração 15: Evolução da temperatura do refrigerador

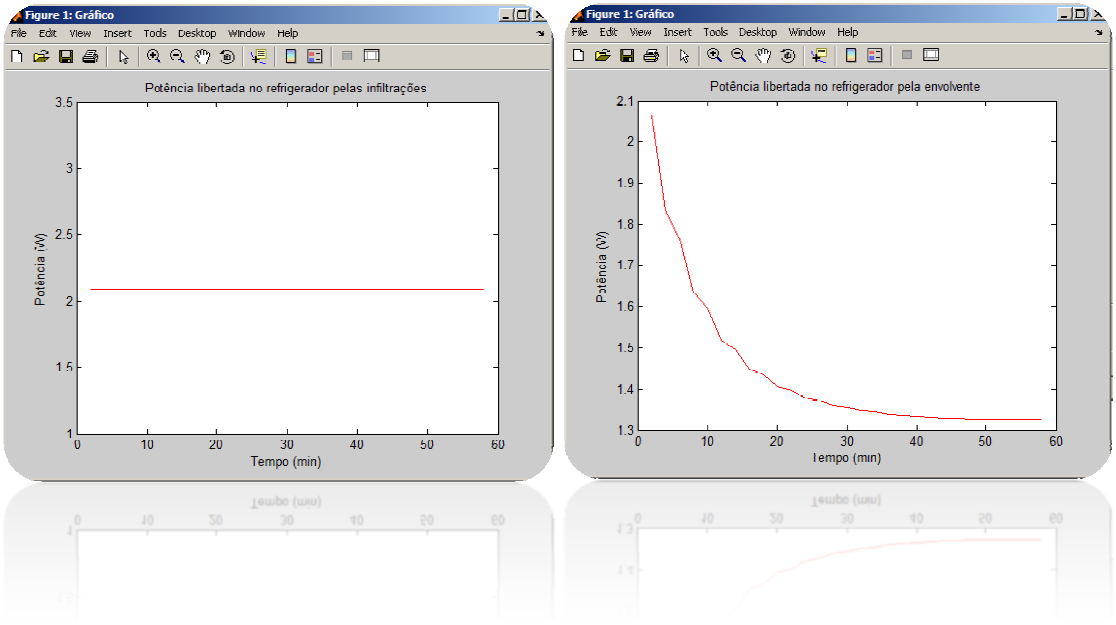


Ilustração 16: Potências libertadas no refrigerador

A temperatura do refrigerador, no período simulado, aumenta exponencialmente, até atingir uma aparente temperatura de equilíbrio, por volta dos 10ºC. Este aumento reflecte a entrada de potências tanto por parte dos produtos perecíveis, como por parte da envolvente, das infiltrações e das renovações de ar, como demonstrado nas figuras acima.

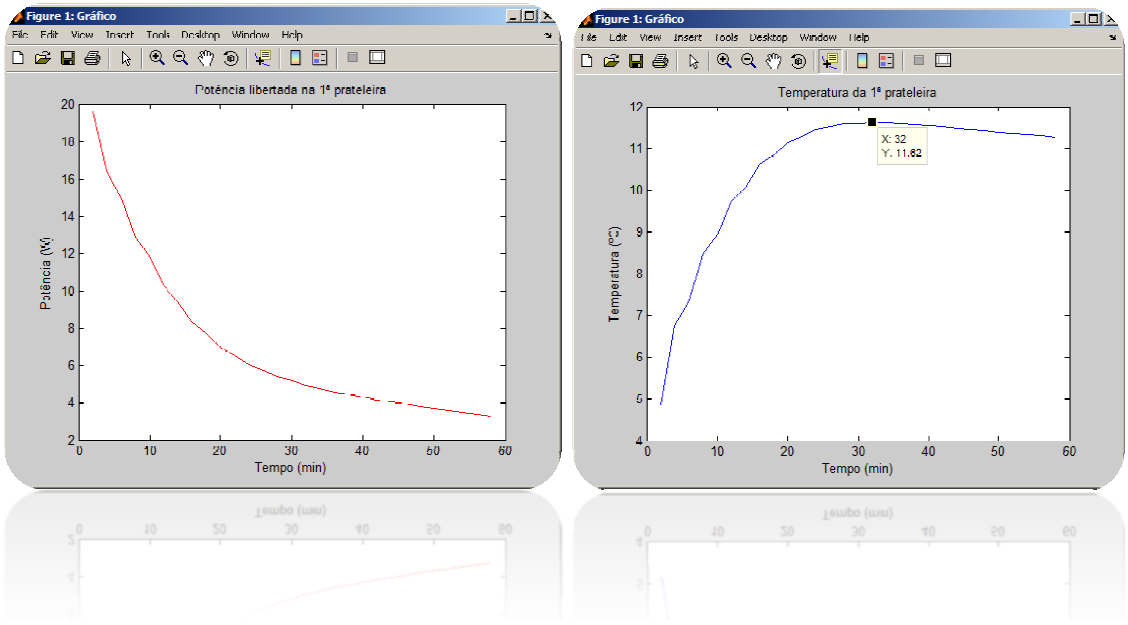


Ilustração 17. Resultados obtidos para a 1ª prateleira

Na 1ª prateleira é possível observar, na Ilustração 17 que a temperatura máxima devida a este carregamento simulado já foi atingida, sendo o seu valor de 11,62ºC, iniciando aos 32 minutos da simulação a descida de temperatura devida à menor libertação de calor por parte dos produtos e à acção do evaporador. As figuras exibidas na ilustração 17 são típicas do que esperaria desta simulação, em que a potência libertada é maior nos primeiros minutos de simulação, sendo a acentuada variação verificada devida à grande diferença de temperaturas, que acarreta uma maior transferência de calor sensível e à respiração e transpiração dos produtos, que implica, como mencionado anteriormente, no início da refrigeração, uma maior transmissão de calor latente.

O mesmo se verifica nas restantes localizações, como se pode observar pelos gráficos das respectivas potências recebidas e evolução de temperatura.

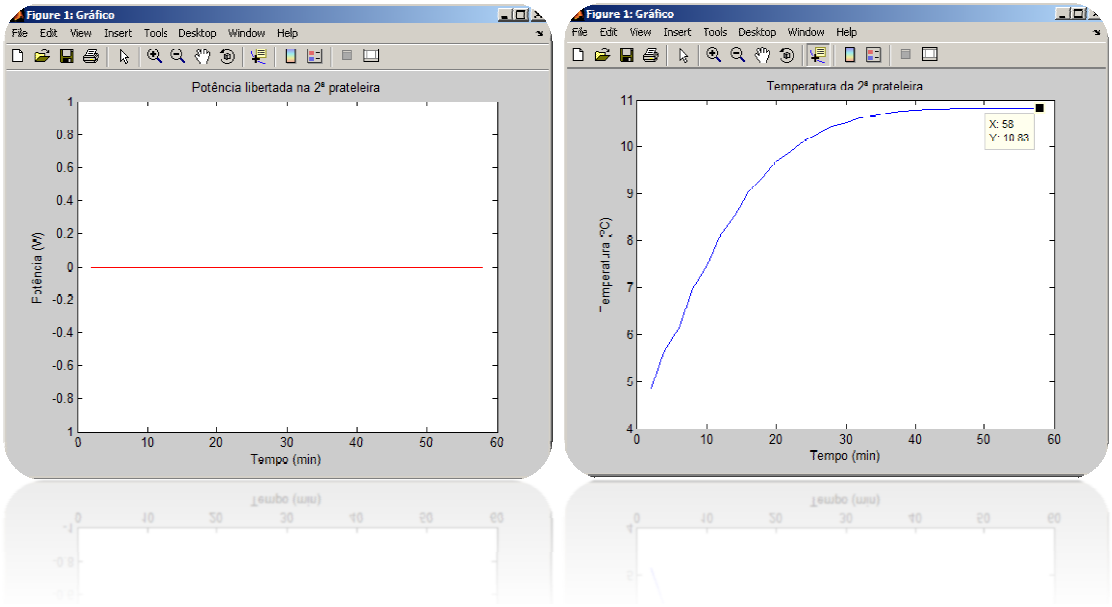


Ilustração 18: Resultados obtidos para a 2ª prateleira

Na 2ª prateleira, deixada vazia nesta simulação, o aumento da temperatura é devido às cargas térmicas recebidas por radiação e convecção das localizações à sua volta. Não tendo inércia, que um produto à temperatura do refrigerador poderia oferecer, para as atenuar, a sua temperatura vai variar de acordo com as variações das cargas térmicas emitidas pela 1ª e 3ª prateleiras.

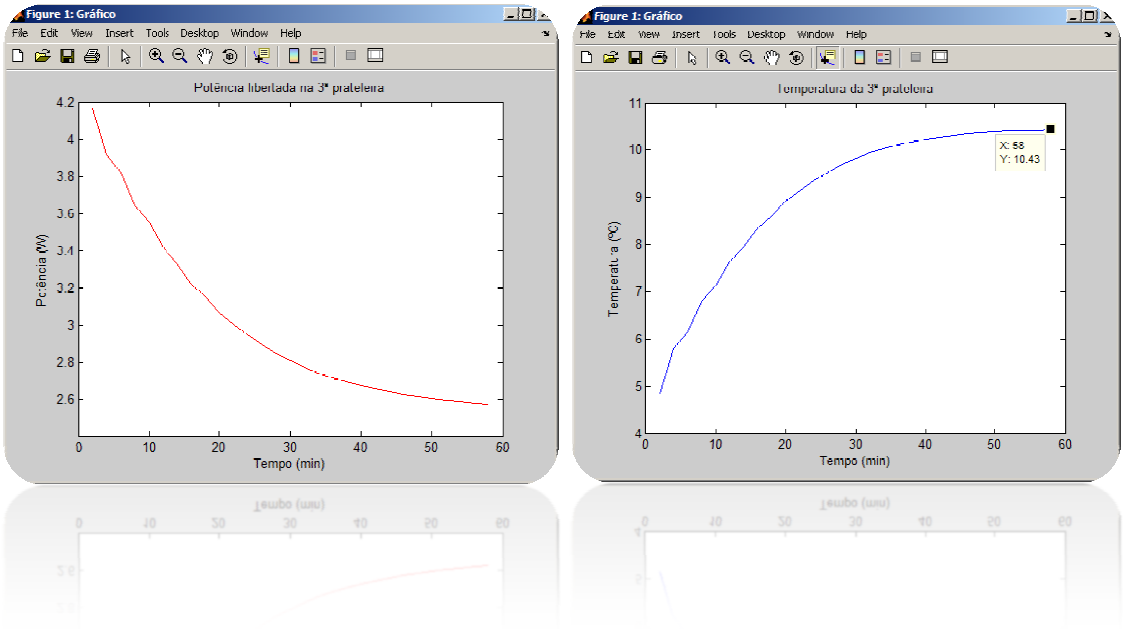


Ilustração 19: Resultados obtidos para a 3ª prateleira

A evolução, tanto da potência recebida, como da variação da temperatura da 3ª prateleira é menos acentuada do que as variações observadas na 1ª prateleira. Isto

indicia a presença de uma maior inércia térmica nesta localização, aqui proporcionada pelo armazenamento de uma melancia.

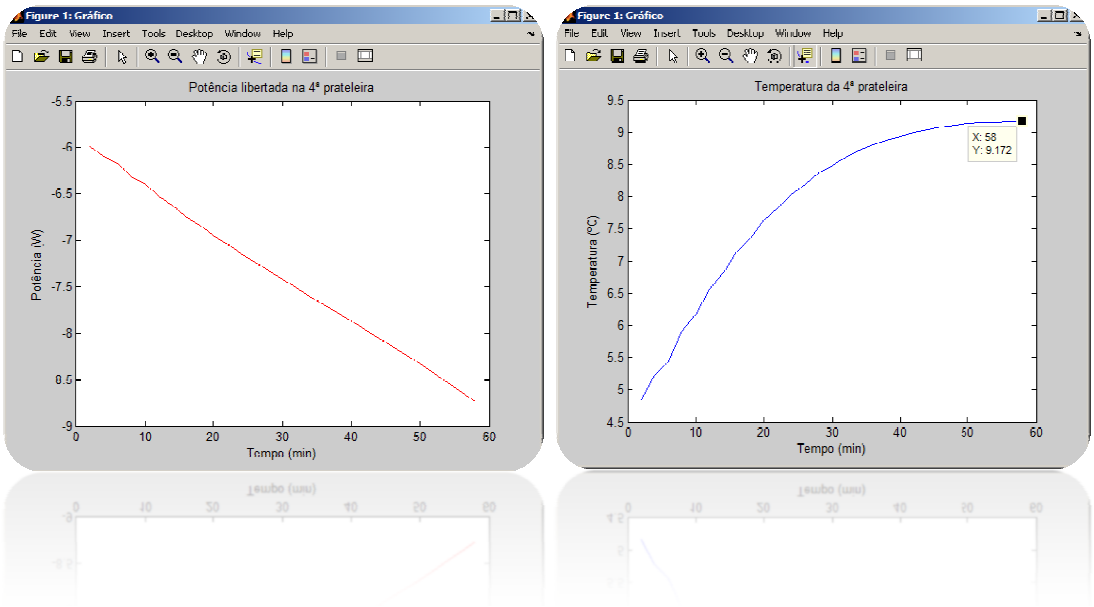


Ilustração 20: Resultados obtidos para a 4ª prateleira

A quarta prateleira tem uma evolução diferente das anteriores, pois trata-se do descongelamento de pescadas, e observa-se uma potência absorvida, neste período, crescente, fruto do aumento da temperatura no refrigerador e também devida às complexas trocas de calor das restantes prateleiras e porta, com o ar em volta desta localização. Esta absorção de potência é, no entanto, insuficiente para evitar a tendência crescente da temperatura apresentada, estando a temperatura da 4ª prateleira a aproximar-se dos 9,17ºC.

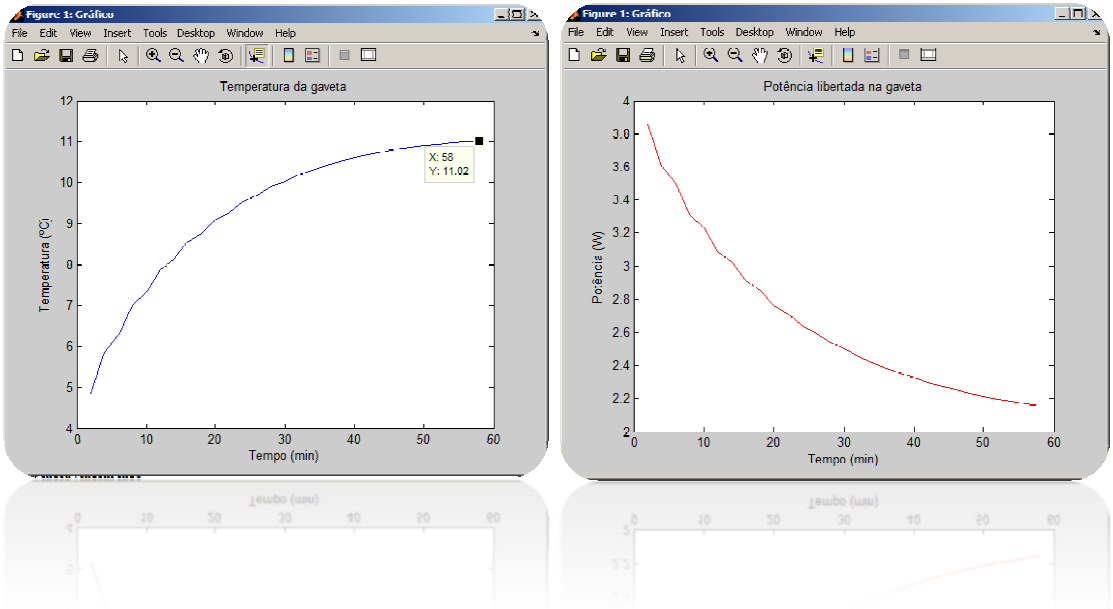


Ilustração 21: Resultados obtidos para a gaveta

Nas gavetas ainda não foi atingida a estabilização da temperatura, que continua, ao fim de 1 hora, a aumentar, devida, não só às cargas libertadas nesta localização, mas também pelas libertadas na porta pelas garrafas, que se encontram muito próximas.

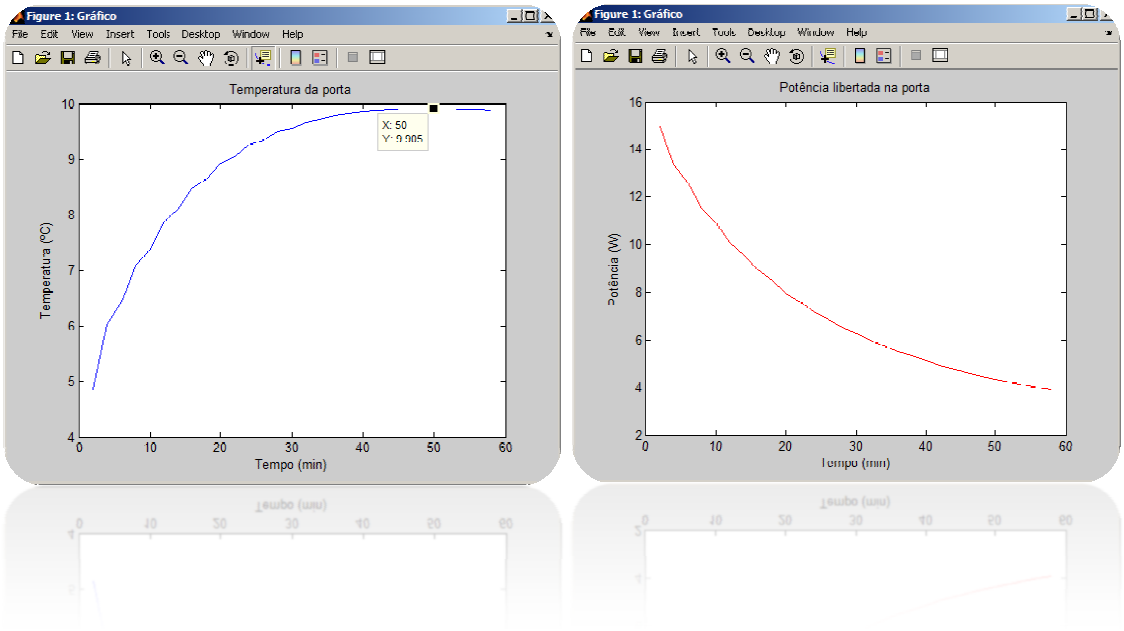


Ilustração 22: Resultados obtidos para a porta

Em todas as localizações é também perceptível a maior dinâmica inicial de fenómenos térmicos, observável nas figuras pela ondulação inicial das linhas de temperatura e potência.

Para completar a análise anterior são apresentadas, em seguida, as evoluções das temperaturas dos produtos no frigorífico.

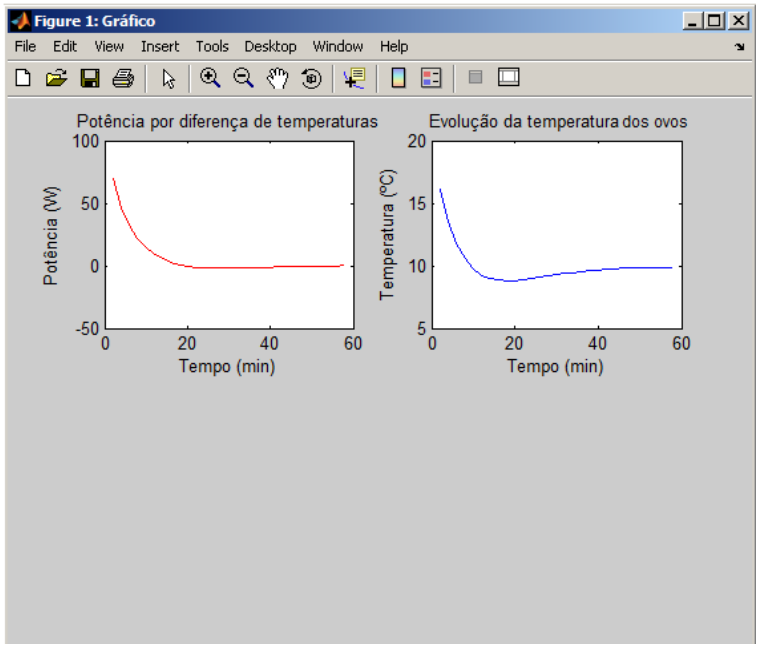


Ilustração 23: Resultados obtidos para os ovos

Os ovos têm uma evolução atípica, uma vez que, devido à sua pequena inércia térmica arrefecem rapidamente e verifica-se o seu arrefecimento até uma temperatura abaixo da da sua localização, absorvendo, a partir daí, potência, até chegar ao equilíbrio. Este comportamento pode ser devido ao passo temporal escolhido para a simulação, em que, em dois minutos, a temperatura dos ovos estaria próxima mas superior à da sua envolvente, e na iteração seguinte, baixou para se aproximar ou mesmo igualar a temperatura da envolvente anterior, mas esta, devido às restantes potências térmicas, subiu, fazendo com que os ovos estivessem, nesse passo, a uma temperatura inferior à da sua localização e consequentemente, deu-se a absorção de potência verificada.

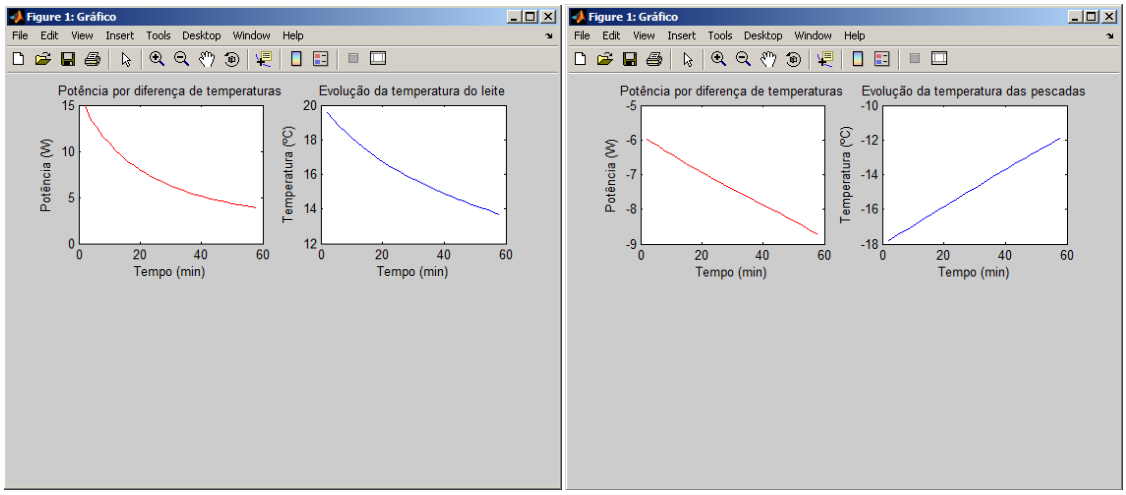


Ilustração 24: Resultados obtidos para o leite e pescadas

O gráfico de libertação de potência do leite mostra que é este produto que define a variação da potência na porta, o que se pode verificar comparando as figuras das duas localizações.

A temperatura das pescadas apresenta, também como esperado, uma constante subida, apresentando uma variação quase linear, indicando a grande inércia inicial do produto em descongelamento.

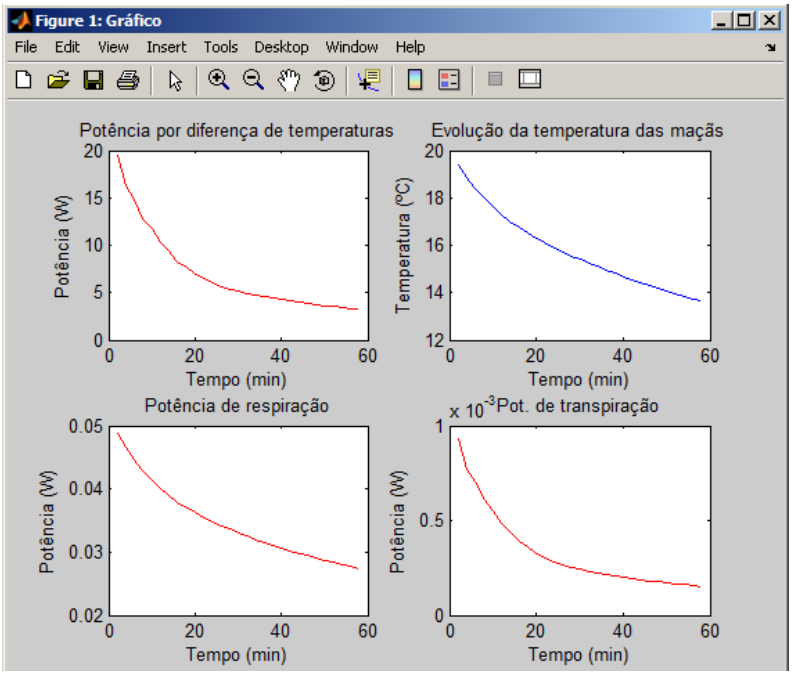


Ilustração 25: Resultados obtidos para as maçãs

A Ilustração 25 mostra as variações típicas de frutos de pequena dimensão, em que não existe uma grande inércia térmica, pelo que a variação de temperatura é mais acentuada no início da refrigeração, quando existe uma maior diferença entre as temperaturas do produto e da envolvente, pelo que o calor sensível trocado é maior. Nesta ilustração são também mostradas as potências libertadas por respiração e transpiração, que decrescem com o aumento da temperatura da envolvente, como seria esperado.

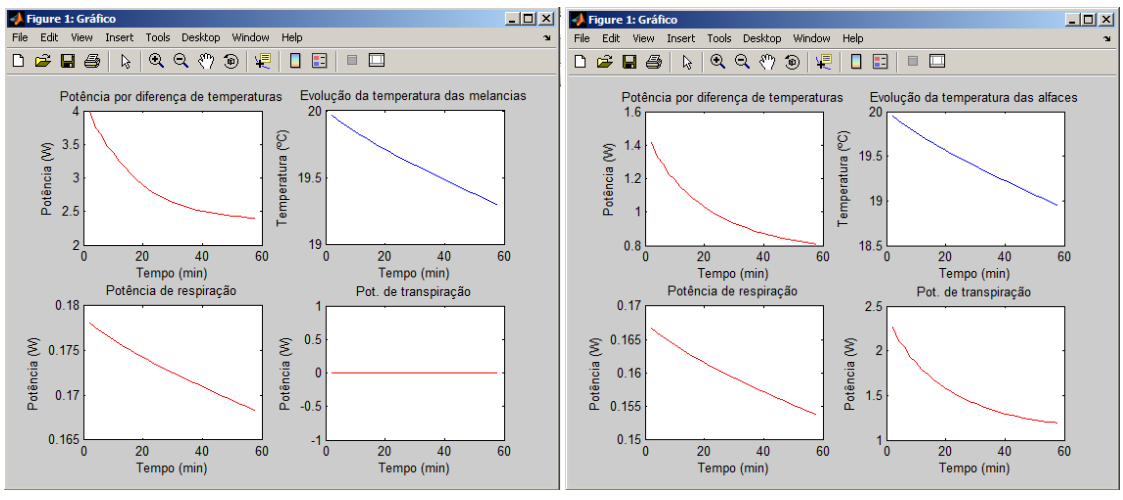


Ilustração 26: Resultados obtidos para as melancias e para as alfaces

Na Ilustração 26 não existe potência libertada por transpiração para as melancias porque não foi encontrada informação disponível acerca desta potência para estas. Ambas as figuras mostram que os produtos a que se referem possuem grandes inércias térmicas,

sendo muito pequena a sua variação de temperatura, mesmo no início da refrigeração, quando existe maior libertação de potências devidas à maior diferença de temperaturas, potências de respiração e de transpiração.

A Ilustração 27, do estado do ar interior do frigorífico, foi retirada do FLUENT e apresenta as temperaturas nas diversas localizações do frigorífico, ao fim de 1 hora.

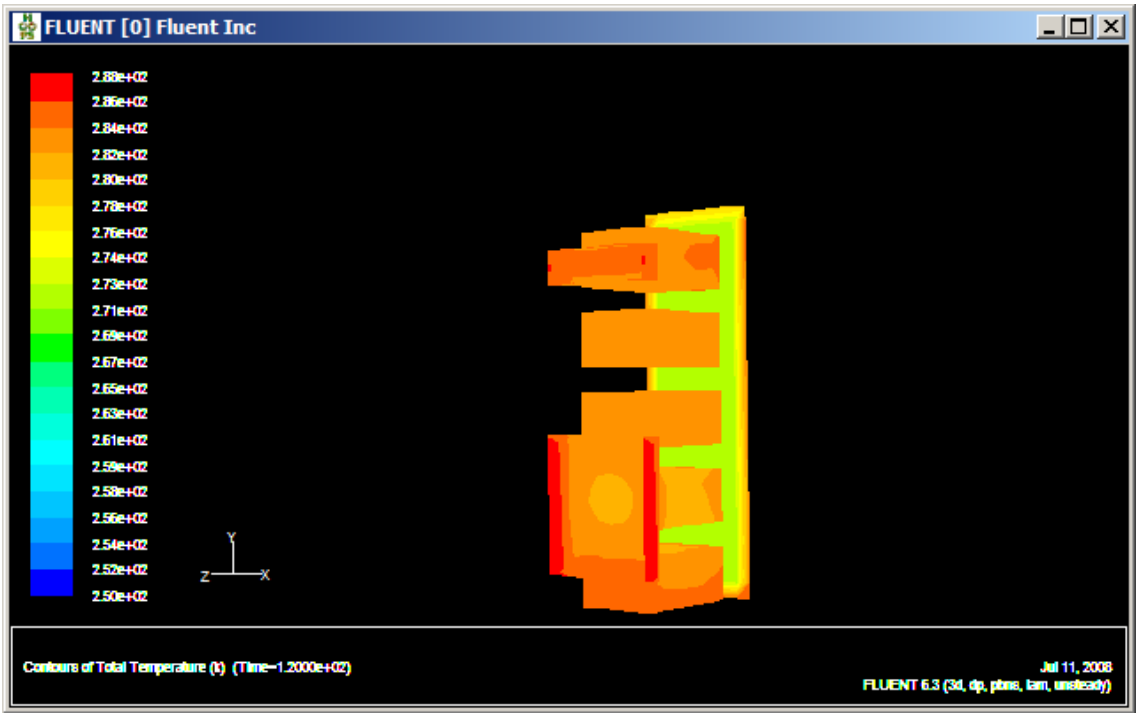


Ilustração 27. Visualização das temperaturas do ar interior do frigorífico

Nesta figura pode-se observar o que já foi mencionado anteriormente, como a harmonização da temperatura da 2ª prateleira com as prateleiras que lhe estão mais próximas e a influência das garrafas na 4ª prateleira e vice-versa, pois a temperatura das garrafas está a diminuir, desde o seu centro, onde a influência das pescadas a descongelar é maior, para a periferia.

6. CONCLUSÕES E PERSPECTIVAS DE TRABALHO FUTURO

6.1. Conclusões

Embora não tenham sido realizadas experiências para confirmar os valores calculados pelo programa, os resultados obtidos para a libertação de calor pelos produtos perecíveis são aqueles que se esperariam, tendencialmente decrescentes, uma vez que a maior taxa de libertação de energia se dá no início da refrigeração/congelamento, existindo períodos de abrandamento destes valores, devidos às variações da temperatura do ar envolvente dos produtos.

As temperaturas obtidas seguem também padrões esperados, verificando-se, no geral, uma maior variação de temperatura nos primeiros minutos, devido à maior transferência de potência. As variações mais lineares de temperaturas devem-se a tempos de refrigeração muito grandes, devidos à grande inércia térmica de alguns produtos, como, por exemplo, a melancia, que devido ao seu grande volume demora muito tempo até o seu centro térmico atingir a temperatura do refrigerador (ou 0,5ºC acima desta temperatura, como foi escolhido neste programa). Estes grandes tempos de refrigeração fazem com que não se consiga, na primeira hora de análise, ver o padrão típico de arrefecimento do produto.

6.2. Perspectivas de trabalho futuro

Para trabalho futuro, fica o melhoramento de algumas capacidades e funções do programa, que, por limitações de tempo, não puderam ser incluídas nesta versão. É desejável o conhecimento e inserção de mais produtos de origem portuguesa, para simular os produtos que se encontram nos frigoríficos portugueses, também é necessário o cálculo do coeficiente de transferência de calor à superfície de mais produtos, pois, para este cálculo, foi considerado um coeficiente para cada tipo de produtos, baseado em tamanhos predominantes de elementos desse tipo, e não para os produtos e suas formas individuais. Uma outra funcionalidade que não foi possível completar prende-se com a avaliação dos tempos de refrigeração, que, para produtos cujo peso seja menor que o peso individual, ou cuja quantidade seja menor que a unidade, não actualiza ainda as dimensões dessa porção do produto, aumentando assim o tempo de refrigeração em relação ao real. Também a validação dos resultados deste programa através de experimentação seria uma perspectiva a concretizar.

Relativamente à simulação com o FLUENT é necessária uma revisão das malhas criadas para que sejam refinadas nas diversas localizações do frigorífico onde podem ser colocados os produtos, sendo também um objectivo correr a simulação sempre na mesma janela do FLUENT, o que diminuiria bastante os tempos de cálculo.

Um outro melhoramento a fazer ao programa é a simulação do ciclo de arrefecimento do frigorífico, modelando o conjunto evaporador-compressor-condensador-válvula de expansão, para uma maior proximidade às condições reais de arrefecimento.

7. REFERÊNCIAS

[3]Ballard, R. N.; “Calculating refrigeration loads on an hour-by-hour basis: part II”; ASHRAE Transactions.

Cecchini, C., and D. Marchal; 1991; “A simulation model of refrigerating and air-conditioning equipment based on experimental data”; ASHRAE Transactions 97(2).

[2]Chaves, António J. Leal; 2006; “Viva melhor”; Edições Une.

Cleland, Earle; 1987; “Prediction of freezing and thawing times for multi-dimensional shapes by simple formulae – Part I: Regular shapes”; International Journal of refrigeration 10(3).

Cleland, Earle; 1987; “Prediction of freezing and thawing times for multi-dimensional shapes by simple formulae – Part II: Irregular shapes”; International Journal of refrigeration 10(4).

[4] Clito Afonso, Manuel Castro e Joaquim Matos; 2007; “Air infiltration on domestic refrigerators: the influence of magnetic seals”.

[5]Clito Afonso; “Refrigeração”; 2006; Faculdade de engenharia da Universidade do Porto.

Rohsenow, W. M.; 1973; “Handbook of heat transfer”; MacGraw Hill.

[1] Vários; 2002; ASHRAE Refrigeration.

ANEXOS

ANEXO A: FUNÇÕES

```
function
[t_parede_calc,Qp,t_parede_int,alfa_conv_i,alfa_conv_e]=alfas(temp_amb
,temp_r,temp_c,alt,larg,prof,topo_baixo,paredes,ref_cong,esp_ext,esp_i
sol,esp_int)

%Esta função determina as temperaturas das paredes exteriores e
interiores do refrigerador ou do congelador, em ºC, para servirem como
entrada às funções de cálculo das potências introduzidas pela
envolvente. Tem como entradas as temperaturas ambiente, do
refrigerador e do congelador, em ºC; a altura, largura e profundidade,
em m, das paredes;  a indicação de se tratar do refrigerador ou do
congelador e as indicações de se tratar de paredes laterais, de topo
ou de fundo.

t_parede(1)=-10; t_parede(2)=temp_amb-5; n=2;

if ref_cong==1
    temp_int=temp_r;
else
    temp_int=temp_c;
end

[ro,cp,cond,visc_din,visc_cin,beta,D,Pr]=propriedad_ar(temp_amb);
[ro_i,cp_i,cond_i,visc_din_i,visc_cin_i,beta_i,D_i,Pr_i]=propriedad_ar
(temp_int);

while abs(t_parede(n)-t_parede(n-1))>0.1

    if paredes==1
    Gr=(ro^2*9.81*beta*(temp_amb-t_parede(n))*alt^3)/(visc_din^2);
    Ra=Gr*Pr;
    Gr_i=(ro_i^2*9.81*beta_i*5*alt^3)/(visc_din_i^2); %5 é o delta_T;
    Ra_i=Gr_i*Pr_i;
        if Ra<1e+9
            Nu=0.59*Ra^0.25; alfa_conv_e=Nu*alt/cond;
            Nu_i=0.59*Ra_i^0.25; alfa_conv_i=Nu_i*alt/cond_i;
        else
            Nu=0.1*Ra^(1/3); alfa_conv_e=Nu*alt/cond;
            Nu_i=0.1*Ra_i^(1/3); alfa_conv_i=Nu_i*alt/cond_i;
        end
    end
    if paredes==2
        if topo_baixo==1
        L_ref=larg*prof/(2*larg+2*prof); Gr=(ro^2*9.81*beta*(temp_amb-
t_parede(n))*L_ref^3)/(visc_din^2); Ra=Gr*Pr;
        Gr_i=(ro^2*9.81*beta_i*5*L_ref^3)/(visc_din_i^2);
        Ra_i=Gr_i*Pr_i;
        if Ra<1e+7
            Nu=0.54*Ra^0.25; alfa_conv_e=Nu*L_ref/cond;
            Nu_i=0.54*Ra_i^0.25; alfa_conv_i=Nu_i*L_ref/cond_i;
        else
            Nu=0.15*Ra^(1/3); alfa_conv_e=Nu*L_ref/cond;
            Nu_i=0.15*Ra_i^(1/3); alfa_conv_i=Nu_i*L_ref/cond_i;
        end
        end
        if topo_baixo==2
        L_ref=larg*prof/(2*larg+2*prof); Gr=(ro^2*9.81*beta*(temp_amb-
t_parede(n))*L_ref^3)/(visc_din^2);
        Ra=Gr*Pr; Nu=0.27*Ra^0.25; alfa_conv_e=Nu*L_ref/cond;
```

```
        Gr_i=(ro_i^2*9.81*beta_i*5*L_ref^3)/(visc_din_i^2);
        Ra_i=Gr_i*Pr_i; Nu_i=0.27*Ra_i^0.25;
        alfa_conv_i=Nu_i*L_ref/cond_i;
        end
    end

    lambda_ext=45; lambda_is=0.027; lambda_int=0.52;

    R_conv_ed_e=1/(alfa_conv_e*alt*prof);
    R_cond_e=esp_ext/(lambda_ext*alt*prof);
    R_cond_is=esp_isol/(lambda_is*alt*prof);
    R_cond_int=esp_int/(lambda_int*alt*prof);
    R_conv_ed_i=1/(alfa_conv_i*alt*prof);

    R_total=R_conv_ed_e+R_cond_e+R_cond_is+R_cond_int+R_conv_ed_i;
    U=1/R_total;
    Qp=U*alt*prof*(temp_amb-temp_int); n=n+1;
    t_parede(n)=temp_amb-Qp*R_conv_ed_e;
end
    t_parede_calc=t_parede(n);
    t_parede_int=temp_int+Qp*R_conv_ed_i;
end
```

```
function
[qp_er,qp_dr,qp_fr]=calor_paredes_r(temp_amb,temp_r,temp_c,refr_cong,a
lt,larg,prof,esp_ext,esp_isol,esp_int)

%esta função determina a potência térmica, em W, que atravessa as
paredes do refrigerador. Considera como entradas as temperaturas
ambiente, do refrigerador, do congelador e as temperaturas das paredes
laterais, de topo e de fundo, em ºC, a altura, largura e profundidade,
%em m, das paredes do refrigerador.

if refr_cong==1

[temp_p_ed,Q,temp_p_ed_i]=alfas(temp_amb,temp_r,temp_c,alt,larg,prof,1
,1,1,esp_ext,esp_isol,esp_int);
else
[temp_p_ed,Q,temp_p_ed_i]=alfas(temp_amb,temp_r,temp_c,alt,larg,prof,1
,1,2,esp_ext,esp_isol,esp_int);
end

[ro,cp,cond,visc_din,visc_cin,beta,D,Pr]=propriedad_ar(temp_amb);
[ro_i,cp_i,cond_i,visc_din_i,visc_cin_i,beta_i,D_i,Pr_i]=propriedad_ar
(temp_r);

lambda_ext=45; lambda_is=0.027; lambda_int=0.52; epsilon=1;

%Refrigerador
%Paredes esquerda e direita:
alfa_rad_e_ed=epsilon*5.729e-
8*(temp_p_ed+temp_amb)*(temp_p_ed^2+temp_amb^2);
alfa_rad_i_ed=epsilon*5.729e-
8*(temp_p_ed_i+temp_r)*(temp_p_ed_i^2+temp_r^2);
Gr_ed=(ro^2*9.81*beta*(temp_amb-temp_p_ed)*alt^3)/(visc_din^2);
Ra_ed=Gr_ed*Pr;
Gr_i_ed=(ro_i^2*9.81*beta_i*5*alt^3)/(visc_din_i^2);
Ra_i_ed=Gr_i_ed*Pr_i;
if Ra_ed<1e+9
    Nu_ed=0.59*Ra_ed^0.25; alfa_conv_e_ed=Nu_ed*alt/cond;
    Nu_i_ed=0.59*Ra_i_ed^0.25; alfa_conv_i_ed=Nu_i_ed*alt/cond_i;
else
    Nu_ed=0.1*Ra_ed^(1/3); alfa_conv_e_ed=Nu_ed*alt/cond;
    Nu_i_ed=0.1*Ra_i_ed^(1/3); alfa_conv_i_ed=Nu_i_ed*alt/cond_i;
end

R_conv_ed_e=1/(alfa_conv_e_ed*alt*prof);
R_rad_ed_e=1/(alfa_rad_e_ed*alt*prof);
R_cond_e=esp_ext/(lambda_ext*alt*prof);
R_cond_is=esp_isol/(lambda_is*alt*prof);
R_cond_int=esp_int/(lambda_int*alt*prof);
R_rad_ed_i=1/(alfa_rad_i_ed*alt*prof);
R_conv_ed_i=1/(alfa_conv_i_ed*alt*prof);

R_total_ed=R_conv_ed_e*R_rad_ed_e/(R_conv_ed_e+R_rad_ed_e)+R_cond_e+R_
cond_is+R_cond_int+(R_conv_ed_i*R_rad_ed_i)/(R_conv_ed_i+R_rad_ed_i);
U_ed=1/R_total_ed;

qp_er=U_ed*(temp_amb-temp_r); qp_dr=U_ed*(temp_amb-temp_r);
qp_fr=U_ed*(temp_amb-temp_r);

%Paredes frente e trás:
```

```
temp_p_ft=alfas(temp_amb,temp_r,temp_c,alt,larg,prof,1,1,1,esp_ext,esp
_isol,esp_int);
temp_p_ft_i=-1.73; alfa_rad_e_ft=epsilon*5.729e-
8*(temp_p_ft+temp_amb)*(temp_p_ft^2+temp_amb^2);
alfa_rad_i_ft=epsilon*5.729e8*(temp_p_ft_i+temp_r)*(temp_p_ft_i^2+temp
_r^2);
Gr_ft=(ro^2*9.81*beta*(temp_amb-temp_p_ft)*alt^3)/(visc_din^2);
Ra_ft=Gr_ft*Pr;
Gr_i_ft=(ro_i^2*9.81*beta_i*5*alt^3)/(visc_din_i^2);
Ra_i_ft=Gr_i_ft*Pr_i;
if Ra_ft<1e+9
    Nu_ft=0.59*Ra_ft^0.25; alfa_conv_e_ft=Nu_ft*alt/cond;
    Nu_i_ft=0.59*Ra_i_ft^0.25; alfa_conv_i_ft=Nu_i_ft*alt/cond_i;
else
    Nu_ft=0.1*Ra_ft^(1/3); alfa_conv_e_ft=Nu_ft*alt/cond;
    Nu_i_ft=0.1*Ra_i_ft^(1/3); alfa_conv_i_ft=Nu_i_ft*alt/cond_i;
end

R_conv_ft_e=1/(alfa_conv_e_ft*alt*larg);
R_rad_ft_e=1/(alfa_rad_e_ft*alt*larg);
R_cond_ft_e=esp_ext/(lambda_ext*alt*larg);
R_cond_ft_is=esp_isol/(lambda_is*alt*larg);
R_cond_ft_int=esp_int/(lambda_int*alt*larg);
R_rad_ft_i=1/(alfa_rad_i_ft*alt*larg);
R_conv_ft_i=1/(alfa_conv_i_ft*alt*larg);

R_total_ft=(R_conv_ft_e*R_rad_ft_e)/(R_conv_ft_e+R_rad_ft_e)+R_cond_ft
_e+R_cond_ft_is+R_cond_ft_int+(R_conv_ft_i*R_rad_ft_i)/(R_conv_ft_i+R_
rad_ft_i);
U_ft=1/R_total_ft;
qp_tr=U_ft*alt*larg*(temp_amb-temp_r);

%Parede de baixo, fundo:
if refr_cong==1
[temp_baixo,Q3,temp_baixo_i]=alfas(temp_amb,temp_r,temp_c,alt,larg,pro
f,2,2,1,esp_ext,esp_isol,esp_int);
else
[temp_baixo,Q3,temp_baixo_i]=alfas(temp_amb,temp_r,temp_c,alt,larg,pro
f,2,2,2,esp_ext,esp_isol,esp_int);
end

alfa_rad_e_b=epsilon*5.729e8*(temp_baixo+temp_amb)*(temp_baixo^2+temp_
amb^2);
alfa_rad_i_b=epsilon*5.729e8*(temp_baixo_i+temp_r)*(temp_baixo_i^2+tem
p_r^2);
L_ref=larg*prof/(2*larg+2*prof);
Gr_b=(ro^2*9.81*beta*(temp_amb-temp_baixo)*L_ref^3)/(visc_din^2);
Ra_b=Gr_b*Pr; Nu_b=0.27*Ra_b^0.25; alfa_conv_e_b=Nu_b*L_ref/cond;
Gr_i_b=(ro_i^2*9.81*beta_i*5*L_ref^3)/(visc_din_i^2);
Ra_i_b=Gr_i_b*Pr_i; Nu_i_b=0.27*Ra_i_b^0.25;
alfa_conv_i_b=Nu_i_b*L_ref/cond_i;

R_conv_b_e=1/(alfa_conv_e_b*prof*larg);
R_rad_b_e=1/(alfa_rad_e_b*prof*larg);
R_cond_b_e=esp_ext/(lambda_ext*prof*larg);
R_cond_b_is=esp_isol/(lambda_is*prof*larg);
R_cond_b_int=esp_int/(lambda_int*prof*larg);
R_rad_b_i=1/(alfa_rad_i_b*prof*larg);
R_conv_b_i=1/(alfa_conv_i_b*prof*larg);
```

```
R_total_b=(R_conv_b_e*R_rad_b_e)/(R_conv_b_e+R_rad_b_e)+R_cond_b_e+R_c
ond_b_is+R_cond_b_int+(R_conv_b_i*R_rad_b_i)/(R_conv_b_i+R_rad_b_i);
U_b=1/R_total_b;

qb=U_b*prof*larg*(temp_amb-temp_r);

%Calor trocado entre as paredes de cima do refrigerador e de baixo do
%congelador.

if refr_cong==1
[temp_topo,Q4,temp_topo_i]=alfas(temp_amb,temp_r,temp_c,alt,larg,prof,
1,2,2,esp_ext,esp_isol,esp_int);
[temp_1,Q5,temp_c_b]=alfas(temp_amb,temp_r,temp_c,alt,larg,prof,1,2,2,
esp_ext,esp_isol,esp_int);
else
[temp_topo,Q4,temp_topo_i]=alfas(temp_amb,temp_r,temp_c,alt,larg,prof,
1,2,2,esp_ext,esp_isol,esp_int);
[temp_1,Q5,temp_c_b]=alfas(temp_amb,temp_r,temp_c,alt,larg,prof,1,2,2,
esp_ext,esp_isol,esp_int);
end

alfa_rad_e_t=epsilon*5.729e8*(temp_c_b+temp_c)*((temp_c_b)^2+(temp_c)^
2);
alfa_rad_i_t=epsilon*5.729e8*(temp_topo_i+temp_r)*(temp_topo_i^2+temp_
r^2); L_ref=larg*prof/(2*larg+2*prof);
Gr_t=(ro^2*9.81*beta*(temp_r-temp_topo_i)*L_ref^3)/(visc_din^2);
Ra_t=Gr_t*Pr; Gr_i_t=(ro^2*9.81*beta_i*5*L_ref^3)/(visc_din_i^2);
Ra_i_t=Gr_i_t*Pr_i;
if Ra_t<1e+7
    Nu_t=0.54*Ra_t^0.25; alfa_conv_e_t=Nu_t*L_ref/cond;
    Nu_i_t=0.54*Ra_i_t^0.25; alfa_conv_i_t=Nu_i_t*L_ref/cond_i;
else
    Nu_t=0.15*Ra_t^(1/3); alfa_conv_e_t=Nu_t*L_ref/cond;
    Nu_i_t=0.15*Ra_i_t^(1/3); alfa_conv_i_t=Nu_i_t*L_ref/cond_i;
end

R_conv_t_e=1/(alfa_conv_e_t*prof*larg);
R_rad_t_e=1/(alfa_rad_e_t*prof*larg);
R_cond_t_e=(2*esp_ext)/(lambda_ext*prof*larg);
R_cond_t_is=(2*esp_isol)/(lambda_is*prof*larg);
R_cond_t_int=(2*esp_int)/(lambda_int*prof*larg);
R_rad_t_i=1/(alfa_rad_i_t*prof*larg);
R_conv_t_i=1/(alfa_conv_i_t*prof*larg);

R_total_t=(R_conv_t_e*R_rad_t_e)/(R_conv_t_e+R_rad_t_e)+R_cond_t_e+R_c
ond_t_is+R_cond_t_int+(R_conv_t_i*R_rad_t_i)/(R_conv_t_i+R_rad_t_i);
U_t=1/R_total_t;
qt=U_t*prof*larg*(temp_amb-temp_c);

end
```

```
function [calor_p_d]=calor_prod_d(tipo,linha,temp_p,massa,local)

%Determina a potência térmica, em W, libertada pelos doces, desde a
sua temperatura inicial até à sua temperatura final. Tem como entradas
o tipo e a linha do produto na matriz dos doces, a sua temperatura, em
ºC, a sua massa, em kg, e o local onde vai ser armazenado. "local" é
pela ordem cong, 1p, 2p, 3p, 4p, gavetas, portas.

tempo=120;
[xm,xp,xfat,xc,xfib,xa,Tempc]=propriedades(tipo,linha);
if temp_p>Tempc
    cp=(caloresp_r(tipo,linha))*1e3;
else
    cp=(caloresp_c(tipo,linha,temp_p))*1e3;
end

if local==1
    temp_c=evalin('base','Temp_c');
    if temp_p>temp_c && temp_p>Tempc
        calor_1=massa*cp*(temp_p-temp_c);calor_2=0;calor_3=0;
    end
    if temp_p<=Tempc && temp_p>temp_c
        calor_1=0;calor_2=0;calor_3=massa*cp*(temp_p-temp_c);
    end
    if temp_p==temp_c || temp_p<temp_c
        calor_1=0;calor_2=0;calor_3=0;
    end
else
    if local==2
            temp_r=evalin('base','Temp_1ap');
    end
    if local==3
            temp_r=evalin('base','Temp_2ap');
    end
    if local==4
            temp_r=evalin('base','Temp_3ap');
    end
    if local==5
            temp_r=evalin('base','Temp_4ap');
    end
    if local==6
            temp_r=evalin('base','Temp_gav');
    end
    if local==7
            temp_r=evalin('base','Temp_porta');
    end
    if temp_p>temp_r || temp_p<temp_r
        calor_1=massa*cp*(temp_p-temp_r);calor_2=0;calor_3=0;
    end
    if temp_p==temp_r
        calor_1=0;calor_2=0;calor_3=0;
    end
end


if linha==1
    calor_p_d=calor_1+calor_2+calor_3;
    if local==1
        tempo_cong=evalin('base','doces_dados(1,40)');
        if temp_p==temp_c
```

```
            tempo_cong=1;
        end
        calor=calor_p_d/tempo_cong;
        nova_temp=temp_p-(calor*tempo/(massa*cp));
        evalin('base',sprintf('c_d(1,i)=%u',calor));
        evalin('base',sprintf('c_d_temp(1,i)=%u',nova_temp));
        evalin('base','clc');
    end
    if local==2
        tempo_refr=evalin('base','doces_dados(1,41)');
        if temp_p==temp_r
            tempo_refr=1;
        end
        calor=calor_p_d/tempo_refr;
        nova_temp=temp_p-calor*tempo/(massa*cp);
        evalin('base',sprintf('r_1ap_d(1,i)=%u',calor));
        evalin('base',sprintf('r_1ap_d_temp(1,i)=%u',nova_temp));
        evalin('base','clc');
     end
     if local==3
        tempo_refr=evalin('base','doces_dados(1,42)');
        if temp_p==temp_r
            tempo_refr=1;
        end
        calor=calor_p_d/tempo_refr;
        nova_temp=temp_p-calor*tempo/(massa*cp);
        evalin('base',sprintf('r_2ap_d(1,i)=%u',calor));
        evalin('base',sprintf('r_2ap_d_temp(1,i)=%u',nova_temp));
        evalin('base','clc');
     end
     if local==4
        tempo_refr=evalin('base','doces_dados(1,43)');
        if temp_p==temp_r
            tempo_refr=1;
        end
        calor=calor_p_d/tempo_refr;
        nova_temp=temp_p-calor*tempo/(massa*cp);
        evalin('base',sprintf('r_3ap_d(1,i)=%u',calor));
        evalin('base',sprintf('r_3ap_d_temp(1,i)=%u',nova_temp));
        evalin('base','clc');
     end
     if local==5
        tempo_refr=evalin('base','doces_dados(1,44)');
        if temp_p==temp_r
            tempo_refr=1;
        end
        calor=calor_p_d/tempo_refr;
        nova_temp=temp_p-calor*tempo/(massa*cp);
        evalin('base',sprintf('r_4ap_d(1,i)=%u',calor));
        evalin('base',sprintf('r_4ap_d_temp(1,i)=%u',nova_temp));
        evalin('base','clc');
     end
end
if linha==2
    calor_p_d=calor_1+calor_2+calor_3;
    if local==1
        tempo_cong=evalin('base','doces_dados(2,40)');
        if temp_p==temp_c
            tempo_cong=1;
        end
        calor=calor_p_d/tempo_cong;
```

```
            nova_temp=temp_p-(calor*tempo/(massa*cp));
            evalin('base',sprintf('c_d(2,i)=%u',calor));
            evalin('base',sprintf('c_d_temp(2,i)=%u',nova_temp));
            evalin('base','clc');
        end
        if local==2
            tempo_refr=evalin('base','doces_dados(2,41)');
            if temp_p==temp_r
                tempo_refr=1;
            end
            calor=calor_p_d/tempo_refr;
            nova_temp=temp_p-calor*tempo/(massa*cp);
            evalin('base',sprintf('r_1ap_d(2,i)=%u',calor));
            evalin('base',sprintf('r_1ap_d_temp(2,i)=%u',nova_temp));
            evalin('base','clc');
         end
         if local==3
            tempo_refr=evalin('base','doces_dados(2,42)');
            if temp_p==temp_r
                tempo_refr=1;
            end
            calor=calor_p_d/tempo_refr;
            nova_temp=temp_p-calor*tempo/(massa*cp);
            evalin('base',sprintf('r_2ap_d(2,i)=%u',calor));
            evalin('base',sprintf('r_2ap_d_temp(2,i)=%u',nova_temp));
            evalin('base','clc');
         end
         if local==4
            tempo_refr=evalin('base','doces_dados(2,43)');
            if temp_p==temp_r
                tempo_refr=1;
            end
            calor=calor_p_d/tempo_refr;
            nova_temp=temp_p-calor*tempo/(massa*cp);
            evalin('base',sprintf('r_3ap_d(2,i)=%u',calor));
            evalin('base',sprintf('r_3ap_d_temp(2,i)=%u',nova_temp));
            evalin('base','clc');
         end
         if local==5
            tempo_refr=evalin('base','doces_dados(2,44)');
            if temp_p==temp_r
                tempo_refr=1;
            end
            calor=calor_p_d/tempo_refr;
            nova_temp=temp_p-calor*tempo/(massa*cp);
            evalin('base',sprintf('r_4ap_d(2,i)=%u',calor));
            evalin('base',sprintf('r_4ap_d_temp(2,i)=%u',nova_temp));
            evalin('base','clc');
         end
    end
    if linha==3
        calor_p_d=calor_1+calor_2+calor_3;
        if local==1
            tempo_cong=evalin('base','doces_dados(3,40)');
            if temp_p==temp_c
                tempo_cong=1;
            end
            calor=calor_p_d/tempo_cong;
            nova_temp=temp_p-(calor*tempo/(massa*cp));
            evalin('base',sprintf('c_d(3,i)=%u',calor));
            evalin('base',sprintf('c_d_temp(3,i)=%u',nova_temp));
```

```
        evalin('base','clc');
    end
    if local==2
        tempo_refr=evalin('base','doces_dados(3,41)');
        if temp_p==temp_r
            tempo_refr=1;
        end
        calor=calor_p_d/tempo_refr;
        nova_temp=temp_p-calor*tempo/(massa*cp);
        evalin('base',sprintf('r_1ap_d(3,i)=%u',calor));
        evalin('base',sprintf('r_1ap_d_temp(3,i)=%u',nova_temp));
        evalin('base','clc');
     end
     if local==3
        tempo_refr=evalin('base','doces_dados(3,42)');
        if temp_p==temp_r
            tempo_refr=1;
        end
        calor=calor_p_d/tempo_refr;
        nova_temp=temp_p-calor*tempo/(massa*cp);
        evalin('base',sprintf('r_2ap_d(3,i)=%u',calor));
        evalin('base',sprintf('r_2ap_d_temp(3,i)=%u',nova_temp));
        evalin('base','clc');
     end
     if local==4
        tempo_refr=evalin('base','doces_dados(3,43)');
        if temp_p==temp_r
            tempo_refr=1;
        end
        calor=calor_p_d/tempo_refr;
        nova_temp=temp_p-calor*tempo/(massa*cp);
        evalin('base',sprintf('r_3ap_d(3,i)=%u',calor));
        evalin('base',sprintf('r_3ap_d_temp(3,i)=%u',nova_temp));
        evalin('base','clc');
     end
     if local==5
        tempo_refr=evalin('base','doces_dados(3,44)');
        if temp_p==temp_r
            tempo_refr=1;
        end
        calor=calor_p_d/tempo_refr;
        nova_temp=temp_p-calor*tempo/(massa*cp);
        evalin('base',sprintf('r_4ap_d(3,i)=%u',calor));
        evalin('base',sprintf('r_4ap_d_temp(3,i)=%u',nova_temp));
        evalin('base','clc');
     end
end
if linha==4
    calor_p_d=calor_1+calor_2+calor_3;
    if local==1
        tempo_cong=evalin('base','doces_dados(4,40)');
        if temp_p==temp_c
            tempo_cong=1;
        end
        calor=calor_p_d/tempo_cong;
        nova_temp=temp_p-(calor*tempo/(massa*cp));
        evalin('base',sprintf('c_d(4,i)=%u',calor));
        evalin('base',sprintf('c_d_temp(4,i)=%u',nova_temp));
        evalin('base','clc');
    end
    if local==2
```

```
            tempo_refr=evalin('base','doces_dados(4,41)');
            if temp_p==temp_r
                tempo_refr=1;
            end
            calor=calor_p_d/tempo_refr;
            nova_temp=temp_p-calor*tempo/(massa*cp);
            evalin('base',sprintf('r_1ap_d(4,i)=%u',calor));
            evalin('base',sprintf('r_1ap_d_temp(4,i)=%u',nova_temp));
            evalin('base','clc');
        end
        if local==3
            tempo_refr=evalin('base','doces_dados(4,42)');
            if temp_p==temp_r
                tempo_refr=1;
            end
            calor=calor_p_d/tempo_refr;
            nova_temp=temp_p-calor*tempo/(massa*cp);
            evalin('base',sprintf('r_2ap_d(4,i)=%u',calor));
            evalin('base',sprintf('r_2ap_d_temp(4,i)=%u',nova_temp));
            evalin('base','clc');
        end
        if local==4
            tempo_refr=evalin('base','doces_dados41,43)');
            if temp_p==temp_r
                tempo_refr=1;
            end
            calor=calor_p_d/tempo_refr;
            nova_temp=temp_p-calor*tempo/(massa*cp);
            evalin('base',sprintf('r_3ap_d(4,i)=%u',calor));
            evalin('base',sprintf('r_3ap_d_temp(4,i)=%u',nova_temp));
            evalin('base','clc');
        end
        if local==5
            tempo_refr=evalin('base','doces_dados(4,44)');
            if temp_p==temp_r
                tempo_refr=1;
            end
            calor=calor_p_d/tempo_refr;
            nova_temp=temp_p-calor*tempo/(massa*cp);
            evalin('base',sprintf('r_4ap_d(4,i)=%u',calor));
            evalin('base',sprintf('r_4ap_d_temp(4,i)=%u',nova_temp));
            evalin('base','clc');
        end
    end
    if linha==5
        calor_p_d=calor_1+calor_2+calor_3;
        if local==1
            tempo_cong=evalin('base','doces_dados(5,40)');
            if temp_p==temp_c
                tempo_cong=1;
            end
            calor=calor_p_d/tempo_cong;
            nova_temp=temp_p-(calor*tempo/(massa*cp));
            evalin('base',sprintf('c_d(5,i)=%u',calor));
            evalin('base',sprintf('c_d_temp(5,i)=%u',nova_temp));
            evalin('base','clc');
        end
        if local==2
            tempo_refr=evalin('base','doces_dados(5,41)');
            if temp_p==temp_r
                tempo_refr=1;
```

```
            end
            calor=calor_p_d/tempo_refr;
            nova_temp=temp_p-calor*tempo/(massa*cp);
            evalin('base',sprintf('r_1ap_d(5,i)=%u',calor));
            evalin('base',sprintf('r_1ap_d_temp(5,i)=%u',nova_temp));
            evalin('base','clc');
        end
        if local==3
            tempo_refr=evalin('base','doces_dados(5,42)');
            if temp_p==temp_r
                tempo_refr=1;
            end
            calor=calor_p_d/tempo_refr;
            nova_temp=temp_p-calor*tempo/(massa*cp);
            evalin('base',sprintf('r_2ap_d(5,i)=%u',calor));
            evalin('base',sprintf('r_2ap_d_temp(5,i)=%u',nova_temp));
            evalin('base','clc');
        end
        if local==4
            tempo_refr=evalin('base','doces_dados(5,43)');
            if temp_p==temp_r
                tempo_refr=1;
            end
            calor=calor_p_d/tempo_refr;
            nova_temp=temp_p-calor*tempo/(massa*cp);
            evalin('base',sprintf('r_3ap_d(5,i)=%u',calor));
            evalin('base',sprintf('r_3ap_d_temp(5,i)=%u',nova_temp));
            evalin('base','clc');
        end
        if local==5
            tempo_refr=evalin('base','doces_dados(5,44)');
            if temp_p==temp_r
                tempo_refr=1;
            end
            calor=calor_p_d/tempo_refr;
            nova_temp=temp_p-calor*tempo/(massa*cp);
            evalin('base',sprintf('r_4ap_d(5,i)=%u',calor));
            evalin('base',sprintf('r_4ap_d_temp(5,i)=%u',nova_temp));
            evalin('base','clc');
        end
    end
    if linha==6
        calor_p_d=calor_1+calor_2+calor_3;
        if local==1
            tempo_cong=evalin('base','doces_dados(6,40)');
            if temp_p==temp_c
                tempo_cong=1;
            end
            calor=calor_p_d/tempo_cong;
            nova_temp=temp_p-(calor*tempo/(massa*cp));
            evalin('base',sprintf('c_d(6,i)=%u',calor));
            evalin('base',sprintf('c_d_temp(6,i)=%u',nova_temp));
            evalin('base','clc');
        end
        if local==2
            tempo_refr=evalin('base','doces_dados(6,41)');
            if temp_p==temp_r
                tempo_refr=1;
            end
            calor=calor_p_d/tempo_refr;
            nova_temp=temp_p-calor*tempo/(massa*cp);
```

```
        evalin('base',sprintf('r_1ap_d(6,i)=%u',calor));
        evalin('base',sprintf('r_1ap_d_temp(6,i)=%u',nova_temp));
        evalin('base','clc');
    end
    if local==3
        tempo_refr=evalin('base','doces_dados(6,42)');
        if temp_p==temp_r
            tempo_refr=1;
        end
        calor=calor_p_d/tempo_refr;
        nova_temp=temp_p-calor*tempo/(massa*cp);
        evalin('base',sprintf('r_2ap_d(6,i)=%u',calor));
        evalin('base',sprintf('r_2ap_d_temp(6,i)=%u',nova_temp));
        evalin('base','clc');
    end
    if local==4
        tempo_refr=evalin('base','doces_dados(6,43)');
        if temp_p==temp_r
            tempo_refr=1;
        end
        calor=calor_p_d/tempo_refr;
        nova_temp=temp_p-calor*tempo/(massa*cp);
        evalin('base',sprintf('r_3ap_d(6,i)=%u',calor));
        evalin('base',sprintf('r_3ap_d_temp(6,i)=%u',nova_temp));
        evalin('base','clc');
    end
    if local==5
        tempo_refr=evalin('base','doces_dados(6,44)');
        if temp_p==temp_r
            tempo_refr=1;
        end
        calor=calor_p_d/tempo_refr;
        nova_temp=temp_p-calor*tempo/(massa*cp);
        evalin('base',sprintf('r_4ap_d(6,i)=%u',calor));
        evalin('base',sprintf('r_4ap_d_temp(6,i)=%u',nova_temp));
        evalin('base','clc');
    end
end
if linha==7
    calor_p_d=calor_1+calor_2+calor_3;
    if local==1
        tempo_cong=evalin('base','doces_dados(7,40)');
        if temp_p==temp_c
            tempo_cong=1;
        end
        calor=calor_p_d/tempo_cong;
        nova_temp=temp_p-(calor*tempo/(massa*cp));
        evalin('base',sprintf('c_d(7,i)=%u',calor));
        evalin('base',sprintf('c_d_temp(7,i)=%u',nova_temp));
        evalin('base','clc');
    end
    if local==2
        tempo_refr=evalin('base','doces_dados(7,41)');
        if temp_p==temp_r
            tempo_refr=1;
        end
        calor=calor_p_d/tempo_refr;
        nova_temp=temp_p-calor*tempo/(massa*cp);
        evalin('base',sprintf('r_1ap_d(7,i)=%u',calor));
        evalin('base',sprintf('r_1ap_d_temp(7,i)=%u',nova_temp));
        evalin('base','clc');
```

```
      end
      if local==3
         tempo_refr=evalin('base','doces_dados(7,42)');
         if temp_p==temp_r
             tempo_refr=1;
         end
         calor=calor_p_d/tempo_refr;
         nova_temp=temp_p-calor*tempo/(massa*cp);
         evalin('base',sprintf('r_2ap_d(7,i)=%u',calor));
         evalin('base',sprintf('r_2ap_d_temp(7,i)=%u',nova_temp));
         evalin('base','clc');
      end
      if local==4
         tempo_refr=evalin('base','doces_dados(7,43)');
         if temp_p==temp_r
             tempo_refr=1;
         end
         calor=calor_p_d/tempo_refr;
         nova_temp=temp_p-calor*tempo/(massa*cp);
         evalin('base',sprintf('r_3ap_d(7,i)=%u',calor));
         evalin('base',sprintf('r_3ap_d_temp(7,i)=%u',nova_temp));
         evalin('base','clc');
      end
      if local==5
         tempo_refr=evalin('base','doces_dados(7,44)');
         if temp_p==temp_r
             tempo_refr=1;
         end
         calor=calor_p_d/tempo_refr;
         nova_temp=temp_p-calor*tempo/(massa*cp);
         evalin('base',sprintf('r_4ap_d(7,i)=%u',calor));
         evalin('base',sprintf('r_4ap_d_temp(7,i)=%u',nova_temp));
         evalin('base','clc');
      end
end
if linha==8
    calor_p_d=calor_1+calor_2+calor_3;
    if local==1
        tempo_cong=evalin('base','doces_dados(8,40)');
        if temp_p==temp_c
            tempo_cong=1;
        end
        calor=calor_p_d/tempo_cong;
        nova_temp=temp_p-(calor*tempo/(massa*cp));
        evalin('base',sprintf('c_d(8,i)=%u',calor));
        evalin('base',sprintf('c_d_temp(8,i)=%u',nova_temp));
        evalin('base','clc');
    end
    if local==2
        tempo_refr=evalin('base','doces_dados(8,41)');
        if temp_p==temp_r
            tempo_refr=1;
        end
        calor=calor_p_d/tempo_refr;
        nova_temp=temp_p-calor*tempo/(massa*cp);
        evalin('base',sprintf('r_1ap_d(8,i)=%u',calor));
        evalin('base',sprintf('r_1ap_d_temp(8,i)=%u',nova_temp));
        evalin('base','clc');
      end
      if local==3
         tempo_refr=evalin('base','doces_dados(8,42)');
```

```
            if temp_p==temp_r
                tempo_refr=1;
            end
            calor=calor_p_d/tempo_refr;
            nova_temp=temp_p-calor*tempo/(massa*cp);
            evalin('base',sprintf('r_2ap_d(8,i)=%u',calor));
            evalin('base',sprintf('r_2ap_d_temp(8,i)=%u',nova_temp));
            evalin('base','clc');
        end
        if local==4
            tempo_refr=evalin('base','doces_dados(8,43)');
            if temp_p==temp_r
                tempo_refr=1;
            end
            calor=calor_p_d/tempo_refr;
            nova_temp=temp_p-calor*tempo/(massa*cp);
            evalin('base',sprintf('r_3ap_d(8,i)=%u',calor));
            evalin('base',sprintf('r_3ap_d_temp(8,i)=%u',nova_temp));
            evalin('base','clc');
        end
        if local==5
            tempo_refr=evalin('base','doces_dados(8,44)');
            if temp_p==temp_r
                tempo_refr=1;
            end
            calor=calor_p_d/tempo_refr;
            nova_temp=temp_p-calor*tempo/(massa*cp);
            evalin('base',sprintf('r_4ap_d(8,i)=%u',calor));
            evalin('base',sprintf('r_4ap_d_temp(8,i)=%u',nova_temp));
            evalin('base','clc');
        end
    end
    if linha==9
        calor_p_d=calor_1+calor_2+calor_3;
        if local==1
            tempo_cong=evalin('base','doces_dados(9,40)');
            if temp_p==temp_c
                tempo_cong=1;
            end
            calor=calor_p_d/tempo_cong;
            nova_temp=temp_p-(calor*tempo/(massa*cp));
            evalin('base',sprintf('c_d(9,i)=%u',calor));
            evalin('base',sprintf('c_d_temp(9,i)=%u',nova_temp));
            evalin('base','clc');
        end
        if local==2
            tempo_refr=evalin('base','doces_dados(9,41)');
            if temp_p==temp_r
                tempo_refr=1;
            end
            calor=calor_p_d/tempo_refr;
            nova_temp=temp_p-calor*tempo/(massa*cp);
            evalin('base',sprintf('r_1ap_d(9,i)=%u',calor));
            evalin('base',sprintf('r_1ap_d_temp(9,i)=%u',nova_temp));
            evalin('base','clc');
        end
        if local==3
            tempo_refr=evalin('base','doces_dados(9,42)');
            if temp_p==temp_r
                tempo_refr=1;
            end
```

```
            calor=calor_p_d/tempo_refr;
            nova_temp=temp_p-calor*tempo/(massa*cp);
            evalin('base',sprintf('r_2ap_d(9,i)=%u',calor));
            evalin('base',sprintf('r_2ap_d_temp(9,i)=%u',nova_temp));
            evalin('base','clc');
        end
        if local==4
            tempo_refr=evalin('base','doces_dados(9,43)');
            if temp_p==temp_r
                tempo_refr=1;
            end
            calor=calor_p_d/tempo_refr;
            nova_temp=temp_p-calor*tempo/(massa*cp);
            evalin('base',sprintf('r_3ap_d(9,i)=%u',calor));
            evalin('base',sprintf('r_3ap_d_temp(9,i)=%u',nova_temp));
            evalin('base','clc');
        end
        if local==5
            tempo_refr=evalin('base','doces_dados(9,44)');
            if temp_p==temp_r
                tempo_refr=1;
            end
            calor=calor_p_d/tempo_refr;
            nova_temp=temp_p-calor*tempo/(massa*cp);
            evalin('base',sprintf('r_4ap_d(9,i)=%u',calor));
            evalin('base',sprintf('r_4ap_d_temp(9,i)=%u',nova_temp));
            evalin('base','clc');
        end
    end
    if linha==10
        calor_p_d=calor_1+calor_2+calor_3;
        if local==1
            tempo_cong=evalin('base','doces_dados(10,40)');
            if temp_p==temp_c
                tempo_cong=1;
            end
            calor=calor_p_d/tempo_cong;
            nova_temp=temp_p-(calor*tempo/(massa*cp));
            evalin('base',sprintf('c_d(10,i)=%u',calor));
            evalin('base',sprintf('c_d_temp(10,i)=%u',nova_temp));
            evalin('base','clc');
        end
        if local==2
            tempo_refr=evalin('base','doces_dados(10,41)');
            if temp_p==temp_r
                tempo_refr=1;
            end
            calor=calor_p_d/tempo_refr;
            nova_temp=temp_p-calor*tempo/(massa*cp);
            evalin('base',sprintf('r_1ap_d(10,i)=%u',calor));
            evalin('base',sprintf('r_1ap_d_temp(10,i)=%u',nova_temp));
            evalin('base','clc');
        end
        if local==3
            tempo_refr=evalin('base','doces_dados(10,42)');
            if temp_p==temp_r
                tempo_refr=1;
            end
            calor=calor_p_d/tempo_refr;
            nova_temp=temp_p-calor*tempo/(massa*cp);
            evalin('base',sprintf('r_2ap_d(10,i)=%u',calor));
```

```
        evalin('base',sprintf('r_2ap_d_temp(10,i)=%u',nova_temp));
        evalin('base','clc');
     end
     if local==4
        tempo_refr=evalin('base','doces_dados(10,43)');
        if temp_p==temp_r
            tempo_refr=1;
        end
        calor=calor_p_d/tempo_refr;
        nova_temp=temp_p-calor*tempo/(massa*cp);
        evalin('base',sprintf('r_3ap_d(10,i)=%u',calor));
        evalin('base',sprintf('r_3ap_d_temp(10,i)=%u',nova_temp));
        evalin('base','clc');
     end
     if local==5
        tempo_refr=evalin('base','doces_dados(10,44)');
        if temp_p==temp_r
            tempo_refr=1;
        end
        calor=calor_p_d/tempo_refr;
        nova_temp=temp_p-calor*tempo/(massa*cp);
        evalin('base',sprintf('r_4ap_d(10,i)=%u',calor));
        evalin('base',sprintf('r_4ap_d_temp(10,i)=%u',nova_temp));
        evalin('base','clc');
     end
end
if linha==11
    calor_p_d=calor_1+calor_2+calor_3;
    if local==1
        tempo_cong=evalin('base','doces_dados(11,40)');
        if temp_p==temp_c
            tempo_cong=1;
        end
        calor=calor_p_d/tempo_cong;
        nova_temp=temp_p-(calor*tempo/(massa*cp));
        evalin('base',sprintf('c_d(11,i)=%u',calor));
        evalin('base',sprintf('c_d_temp(11,i)=%u',nova_temp));
        evalin('base','clc');
    end
    if local==2
        tempo_refr=evalin('base','doces_dados(11,41)');
        if temp_p==temp_r
            tempo_refr=1;
        end
        calor=calor_p_d/tempo_refr;
        nova_temp=temp_p-calor*tempo/(massa*cp);
        evalin('base',sprintf('r_1ap_d(11,i)=%u',calor));
        evalin('base',sprintf('r_1ap_d_temp(11,i)=%u',nova_temp));
        evalin('base','clc');
     end
     if local==3
        tempo_refr=evalin('base','doces_dados(11,42)');
        if temp_p==temp_r
            tempo_refr=1;
        end
        calor=calor_p_d/tempo_refr;
        nova_temp=temp_p-calor*tempo/(massa*cp);
        evalin('base',sprintf('r_2ap_d(11,i)=%u',calor));
        evalin('base',sprintf('r_2ap_d_temp(11,i)=%u',nova_temp));
        evalin('base','clc');
     end
```

```
      if local==4
         tempo_refr=evalin('base','doces_dados(11,43)');
         if temp_p==temp_r
             tempo_refr=1;
         end
         calor=calor_p_d/tempo_refr;
         nova_temp=temp_p-calor*tempo/(massa*cp);
         evalin('base',sprintf('r_3ap_d(11,i)=%u',calor));
         evalin('base',sprintf('r_3ap_d_temp(11,i)=%u',nova_temp));
         evalin('base','clc');
      end
      if local==5
         tempo_refr=evalin('base','doces_dados(11,44)');
         if temp_p==temp_r
             tempo_refr=1;
         end
         calor=calor_p_d/tempo_refr;
         nova_temp=temp_p-calor*tempo/(massa*cp);
         evalin('base',sprintf('r_4ap_d(11,i)=%u',calor));
         evalin('base',sprintf('r_4ap_d_temp(11,i)=%u',nova_temp));
         evalin('base','clc');
      end
end
if linha==13
    calor_p_d=calor_1+calor_2+calor_3;
    if local==1
         tempo_cong=evalin('base','doces_dados(13,40)');
         if temp_p==temp_c
             tempo_cong=1;
         end
         calor=calor_p_d/tempo_cong;
         nova_temp=temp_p-(calor*tempo/(massa*cp));
         evalin('base',sprintf('c_d(13,i)=%u',calor));
         evalin('base',sprintf('c_d_temp(13,i)=%u',nova_temp));
         evalin('base','clc');
    end
    if local==2
         tempo_refr=evalin('base','doces_dados(13,41)');
         if temp_p==temp_r
             tempo_refr=1;
         end
         calor=calor_p_d/tempo_refr;
         nova_temp=temp_p-calor*tempo/(massa*cp);
         evalin('base',sprintf('r_1ap_d(13,i)=%u',calor));
         evalin('base',sprintf('r_1ap_d_temp(13,i)=%u',nova_temp));
         evalin('base','clc');
      end
      if local==3
         tempo_refr=evalin('base','doces_dados(13,42)');
         if temp_p==temp_r
             tempo_refr=1;
         end
         calor=calor_p_d/tempo_refr;
         nova_temp=temp_p-calor*tempo/(massa*cp);
         evalin('base',sprintf('r_2ap_d(13,i)=%u',calor));
         evalin('base',sprintf('r_2ap_d_temp(13,i)=%u',nova_temp));
         evalin('base','clc');
      end
      if local==4
         tempo_refr=evalin('base','doces_dados(13,43)');
         if temp_p==temp_r
```

```
            tempo_refr=1;
        end
        calor=calor_p_d/tempo_refr;
        nova_temp=temp_p-calor*tempo/(massa*cp);
        evalin('base',sprintf('r_3ap_d(13,i)=%u',calor));
        evalin('base',sprintf('r_3ap_d_temp(13,i)=%u',nova_temp));
        evalin('base','clc');
    end
    if local==5
        tempo_refr=evalin('base','doces_dados(13,44)');
        if temp_p==temp_r
            tempo_refr=1;
        end
        calor=calor_p_d/tempo_refr;
        nova_temp=temp_p-calor*tempo/(massa*cp);
        evalin('base',sprintf('r_4ap_d(13,i)=%u',calor));
        evalin('base',sprintf('r_4ap_d_temp(13,i)=%u',nova_temp));
        evalin('base','clc');
    end
end
end
```

```
function [cp_c]= caloresp_c(tipo,linha,temp_p)

%Retorna o valor do calor específico aparente abaixo da temperatura de
%congelamento do produto. Tem como entradas o tipo e linha do produto
na sua respectiva matriz de propriedades e a temperatura, em ºC,
abaixo da sua temperatura inicial de congelamento, a que se encontra.

[xmoist,xprot,xfat,xcarb,xfib,xash,Tempic]=propriedades(tipo,linha);

xb=0.4*xprot; x_sol=1-xmoist;
cp_c=1.55+1.26*x_sol-(xmoist-xb)*333.6*Tempic/(temp_p^2);

end

function [cp_r]= caloresp_r(tipo,linha)
%Retorna o valor do calor específico acima da temperatura de
congelamento do produto. Tem como entrada o tipo e a linha do produto
na respectiva matriz de propriedades.

[xmoist]=propriedades(tipo,linha);
x_sol=1-xmoist;
cp_r=4.19-2.3*x_sol-0.628*x_sol^3;

end

function [kt]=condtermica(tipo, linha, Temp)

%A função determina a condutibilidade térmica dos produtos, em W/m.K à
%temperatura a que se encontram. Tem como entradas o tipo e linha do
produto na respectiva matriz de propriedades e a temperatura a que o
produto se encontra, em ºC.

[xmoist,xprot,xfat,xcarb,xfib,xash,Tempic]=propriedades(tipo,linha);
[ro_t,romoist,roprot,rofat,rocarb,rofib,roash,roice]=densidade(tipo,li
nha,Temp);

if Temp < Tempic
    xice=(1.105*xmoist)/(1+0.8765/log(Tempic-Temp+1));
    xw=xmoist-xice;
else
    xice=0;
    xw=xmoist;
end

k_prot=1.7881e-1+(1.1958e-3)*Temp-(2.7178e-6)*Temp^2;
k_fat=1.8071e-1-(2.7604e-3)*Temp-(1.7749e-7)*Temp^2;
k_carb=2.0141e-1+(1.3874e-3)*Temp-(4.3312e-6)*Temp^2;
k_fib=1.8331e-1+(1.2497e-3)*Temp-(3.1683e-6)*Temp^2;
k_ash=3.2962e-1+(1.4011e-3)*Temp-(2.9069e-6)*Temp^2;
k_ice=2.2196-(6.2489e-3)*Temp+(1.0154e-4)*Temp^2;
k_w=5.7109e-1+(1.7625e-3)*Temp-(6.7036e-6)*Temp^2;

x_totalv=xprot/roprot+xfat/rofat+xcarb/rocarb+xfib/rofib+xash/roash+xw
/romoist+xice/roice;

x_protv=(xprot/roprot)/x_totalv;
x_fatv=(xfat/rofat)/x_totalv;
x_carbv=(xcarb/rocarb)/x_totalv;
x_fibv=(xfib/rofib)/x_totalv;
```

```
x_ashv=(xash/roash)/x_totalv;
x_wv=(xw/romoist)/x_totalv;
x_icev=(xice/roice)/x_totalv;

kt=x_protv*k_prot+x_fatv*k_fat+x_carbv*k_carb+x_fibv*k_fib+x_ashv*k_as
h+x_wv*k_w+x_icev*k_ice;

end
```

%Procedimento checkfiles.m

```
t1=0; t2=0; t3=0; t4=0; t5=0; t6=0; t7=0; t8=0;

while t1==0 || t2==0 || t3==0 || t4==0 || t5==0 || t6==0 || t7==0 ||
t8==0
t1=exist('temp1ap','file'); t2=exist('temp2ap','file');
t3=exist('temp3ap','file'); t4=exist('temp4ap','file');
t5=exist('tempg','file'); t6=exist('tempgarr','file');
t7=exist('temparrefr','file'); t8=exist('temparcong','file');
end

clear Temp_1ap Temp_2ap Temp_3ap Temp_4ap Temp_ gav Temp_porta Temp_r
Temp_c; %Faltam as outras que ainda não estão a ser lidas!
importfile('c:\users\im\documents\matlab\temp1ap');
temp_1ap=(mean(evalin('base','data(:,2)')))-273.15;
evalin('base',sprintf('Temp_1ap=%u',temp_1ap));
!del c:\users\im\documents\matlab\temp1ap
importfile('c:\users\im\documents\matlab\temp2ap');
temp_2ap=(mean(evalin('base','data(:,2)')))-273.15;
evalin('base',sprintf('Temp_2ap=%u',temp_2ap));
!del c:\users\im\documents\matlab\temp2ap
importfile('c:\users\im\documents\matlab\temp3ap');
temp_3ap=(mean(evalin('base','data(:,2)')))-273.15;
evalin('base',sprintf('Temp_3ap=%u',temp_3ap));
!del c:\users\im\documents\matlab\temp3ap
importfile('c:\users\im\documents\matlab\temp4ap');
temp_4ap=(mean(evalin('base','data(:,2)')))-273.15;
evalin('base',sprintf('Temp_4ap=%u',temp_4ap));
!del c:\users\im\documents\matlab\temp4ap
importfile('c:\users\im\documents\matlab\tempg');
temp_gav=(mean(evalin('base','data(:,2)')))-273.15;
evalin('base',sprintf('Temp_gav=%u',temp_gav));
!del c:\users\im\documents\matlab\tempg
importfile('c:\users\im\documents\matlab\tempgarr');
temp_porta=(mean(evalin('base','data(:,2)')))-273.15;
evalin('base',sprintf('Temp_porta=%u',temp_porta));
!del c:\users\im\documents\matlab\tempgarr
importfile('c:\users\im\documents\matlab\temparrefr');
temp_r=(mean(evalin('base','data(:,2)')))-273.15;
evalin('base',sprintf('Temp_r=%u',temp_r));
!del c:\users\im\documents\matlab\temparrefr
importfile('c:\users\im\documents\matlab\temparcong');
temp_c=(mean(evalin('base','data(:,2)')))-273.15;
evalin('base',sprintf('Temp_c=%u',temp_c));
!del c:\users\im\documents\matlab\temparcong
evalin('base','clc');
```

```
function [ro,ro_w,ro_prot,ro_fat,ro_carb,ro_fib,ro_ash,ro_ice]=
densidade(tipo,linha,Temp)

%Calcula a massa volúmica, kg/m3, dos produtos, assim como dos seus
%componentes, para a temperatura a que os produtos se encontram. Tem
como entradas
%o tipo e a linha do produto na respectiva matriz de propriedades e a
sua temperatura, em ºC.

[xmoist,xprot,xfat,xcarb,xfib,xash,Tempic]=propriedades(tipo,linha);

if Temp < Tempic
    x_ice=1.105*xmoist/(1+0.8765/log(Tempic-Temp+1));
    xw=xmoist-x_ice;
else
    x_ice=0;
    xw=xmoist;
end

ro_w=9.9718e2+(3.1439e-3)*Temp-(3.7574e-3)*Temp^2;
ro_prot=1.3299e3-(5.184e-1)*Temp;
ro_fat=9.2559e2-(4.1757e-1)*Temp;
ro_carb=1.5991e3-(3.1046e-1)*Temp;
ro_fib=1.3115e3-(3.6589e-1)*Temp;
ro_ash=2.4238e3-(2.8063e-1)*Temp;
ro_ice=9.1689e2-(1.3071e-1)*Temp;

ro=1/(xw/ro_w+xprot/ro_prot+xfat/ro_fat+xcarb/ro_carb+xfib/ro_fib+xash
/ro_ash+x_ice/ro_ice);

end

function[h_ic,h_c]=entalpia_c(tipo,linha,temp_c)

%Determina a entalpia do produto, em kJ/kg, abaixo da sua temperatura
de congelamento. Tem como entradas o tipo e a linha do produto na
respectiva matriz de propriedades e a sua temperatura, em ºC, abaixo
do valor de congelamento.

[xmoist,xprot,xfat,xcarb,xfib,xash,tempic]=propriedades(tipo,linha);

x_sol=1-xmoist; xb=0.4*xprot;
h_ic=(tempic+40)*(1.55+1.26*x_sol-(xmoist-
xb)*333.6*tempic/(40*tempic));
h_c=(temp_c+40)*(1.55+1.26*x_sol-(xmoist-xb)*333.6*tempic/(-
40*temp_c));

end
```

```
function[h_r]=entalpia_r(tipo,linha,temp_p,temp_c)

%Determina a entalpia dos produtos, em kJ/kg, acima da sua temperatura
de congelamento. Tem como entradas o tipo e a linha do produto na
respectiva matriz de propriedades e as temperaturas do produto e de
congelamento, em ºC.

[xmoist,xprot,xfat,xcarb,xfib,xash,tempic]=propriedades(tipo,linha);
[h_icong,h_cong]=entalpia_c(tipo,linha,temp_c);

x_sol=1-xmoist;
h_r=h_icong+(temp_p-tempic)*(4.19-2.30*x_sol-0.628*x_sol^3);

end

function importfile(fileToRead1)
%Importa os dados do ficheiro fileToRead1.

% Importa o ficheiro.
newData1 = importdata(fileToRead1);

% Cria uma nova variável no espaço de trabalho com os campos lidos.
vars = fieldnames(newData1);
for i = 1:length(vars)
    assignin('base', vars{i}, newData1.(vars{i}));
end
end

function importfile_sim(fileToRead1)
%Importa os dados do ficheiro fileToRead1 com as especificações dadas
de 1053 linhas de cabeçalho, uma vez que se trata de um ficheiro de
dados misto, numérico e
%alfanumérico, onde todas as linhas de dados são tratadas como
cabeçalhos,
%para poderem também ser escritos dados mistos.

DELIMITER = ' ';
HEADERLINES = 1053;

% Importa o ficheiro.
rawData1 = importdata(fileToRead1, DELIMITER, HEADERLINES);

[unused,name] = fileparts(fileToRead1);
newData1.(genvarname(name)) = rawData1;

% Cria uma nova variável no espaço de trabalho com os campos lidos.
vars = fieldnames(newData1);
for i = 1:length(vars)
    assignin('base', vars{i}, newData1.(vars{i}));
end
```

```
function
[Inf,Q_Inf]=infiltr(tempo,novo_velho,cong_ref,vol,temp_amb,temp_r,temp
_c)

%Calcula a taxa de ar infiltrado (s-1) e a respectiva potência, em W,
que entra no frigorífico através das infiltrações que as membranas das
portas deixam passar para o interior, num determinado período de
tempo. Tem como entradas o período de tempo, em minutos, para o qual
se quer calcular a potência infiltrada, a indicação de se tratar de um
frigorífico novo ou velho, com as respectivas membranas a serem
consideradas também, respectivamente, novas ou velhas; a indicação de
o cálculo estar a ser efectuado para o congelador ou para o
refrigerador, o volume de ar existente no local onde se querem
calcular as infiltrações e as temperaturas ambiente, do refrigerador e
do congelador, em ºC.

if cong_ref==2
    temp_int=temp_r;
else
    temp_int=temp_c;
end

[ro,cp_ar]=propriedad_ar(temp_int);

if novo_velho==1
    if cong_ref==2
        ln_c=-2.1227*tempo+7.9683;
    else
        ln_c=-1.107*tempo+7.7824;
    end
else
    if cong_ref==2
        ln_c=-12.661*tempo+13.919;
    else
        ln_c=-6.0419*tempo+11.044;
    end
end

Inf=1/tempo*(6.9078-ln_c);
Q_Inf=vol*ro*Inf*cp_ar*(temp_amb-temp_int);

end

function[ro,h_ar,cp,cond,visc_din,visc_cin,beta,D,Pr]=propriedad_ar(Te
mp)

%esta função calcula as propriedades do ar à sua temperatura. As
%propriedades calculadas são a massa volúmica, em kg/m3, a entalpia,
%em kJ/kg, o calor específico, em kJ/kg.K, a condutibilidade térmica,
%em W/m.K, a viscosidade dinâmica, em N.s/m2, a viscosidade cinemática
em m2/s, o coeficiente beta, em K(-1), e o número de Prandtl. A função
tem como entrada a temperatura do ar, em ºC.

ro_ar=polyfit(evalin('base','propr_ar(:,1)'),evalin('base','propr_ar(:
,2)'),1);
h_curva=polyfit(evalin('base','propr_ar(:,1)'),evalin('base','propr_ar
(:,10)'),1);
cp_ar=polyfit(evalin('base','propr_ar(:,1)'),evalin('base','propr_ar(:
,3)'),1);
```

```
cond_ar=polyfit(evalin('base','propr_ar(:,1)'),evalin('base','propr_ar
(:,4)'),1);
D_ar=polyfit(evalin('base','propr_ar(:,1)'),evalin('base','propr_ar(:,
8)'),1);
visc_din_ar=polyfit(evalin('base','propr_ar(:,1)'),evalin('base','prop
r_ar(:,5)'),1);
visc_cin_ar=polyfit(evalin('base','propr_ar(:,1)'),evalin('base','prop
r_ar(:,6)'),1);
beta_ar=polyfit(evalin('base','propr_ar(:,1)'),evalin('base','propr_ar
(:,7)'),1);
Pr_ar=polyfit(evalin('base','propr_ar(:,1)'),evalin('base','propr_ar(:
,9)'),1);

ro=polyval(ro_ar,Temp);
h_ar=polyval(h_curva,Temp);
cp=polyval(cp_ar,Temp);
cond=polyval(cond_ar,Temp);
visc_din=polyval(visc_din_ar,Temp);
visc_cin=polyval(visc_cin_ar,Temp);
beta=polyval(beta_ar,Temp);
D=polyval(D_ar,Temp);
Pr=polyval(Pr_ar,Temp);

end

function[x_moist,x_prot,x_fat,x_carb,x_fib,x_ash,Temp_ic,cp_ref,cp_con
g,calor_lat,peso,maior_d,menor_d,menor_d2,dim_caract,lambda,gama_1,gam
a_2,N,p1,p2,p3,G1,G2,G3]=propriedades(tipo,linha)
%A função devolve as propriedades de um determinado produto, e os seus
%componentes. Tem como entrada o tipo e a linha do produto na
respectiva matriz de propriedades.

load Propriedades_Bebidas.mat; load Propriedades_Carnes.mat;
load Propriedades_Doces.mat; load Propriedades_Frutos.mat;
load Propriedades_Lacticinios.mat; load Propriedades_Peixes.mat;
load Propriedades_Vegetais.mat;

if tipo==1
x_moist=beb_dados(linha,1);
x_prot=beb_dados(linha,2);
x_fat=beb_dados(linha,3);
x_carb=beb_dados(linha,4);
x_fib=beb_dados(linha,5);
x_ash=beb_dados(linha,6);
Temp_ic=beb_dados(linha,7);
cp_ref=beb_dados(linha,8);
cp_cong=beb_dados(linha,9);
calor_lat=beb_dados(linha,10);
peso=beb_dados(linha,25);
maior_d=beb_dados(linha,26);
menor_d=beb_dados(linha,27);
menor_d2=beb_dados(linha,28);
dim_caract=beb_dados(linha,29);
lambda=beb_dados(linha,30);
gama_1=beb_dados(linha,31);
gama_2=beb_dados(linha,32);
N=beb_dados(linha,33);
p1=beb_dados(linha,34);
p2=beb_dados(linha,35);
p3=beb_dados(linha,36);
```

```
G1=beb_dados(linha,37);
G2=beb_dados(linha,38);
G3=beb_dados(linha,39);
end

if tipo==2
x_moist=carne_dados(linha,1);
x_prot=carne_dados(linha,2);
x_fat=carne_dados(linha,3);
x_carb=carne_dados(linha,4);
x_fib=carne_dados(linha,5);
x_ash=carne_dados(linha,6);
Temp_ic=carne_dados(linha,7);
cp_ref=carne_dados(linha,8);
cp_cong=carne_dados(linha,9);
calor_lat=carne_dados(linha,10);
peso=carne_dados(linha,25);
maior_d=carne_dados(linha,26);
menor_d=carne_dados(linha,27);
menor_d2=carne_dados(linha,28);
dim_caract=carne_dados(linha,29);
lambda=carne_dados(linha,30);
gama_1=carne_dados(linha,31);
gama_2=carne_dados(linha,32);
N=carne_dados(linha,33);
p1=carne_dados(linha,34);
p2=carne_dados(linha,35);
p3=carne_dados(linha,36);
G1=carne_dados(linha,37);
G2=carne_dados(linha,38);
G3=carne_dados(linha,39);
end

if tipo==3
x_moist=doces_dados(linha,1);
x_prot=doces_dados(linha,2);
x_fat=doces_dados(linha,3);
x_carb=doces_dados(linha,4);
x_fib=doces_dados(linha,5);
x_ash=doces_dados(linha,6);
Temp_ic=doces_dados(linha,7);
cp_ref=doces_dados(linha,8);
cp_cong=doces_dados(linha,9);
calor_lat=doces_dados(linha,10);
peso=doces_dados(linha,25);
maior_d=doces_dados(linha,26);
menor_d=doces_dados(linha,27);
menor_d2=doces_dados(linha,28);
dim_caract=doces_dados(linha,29);
lambda=doces_dados(linha,30);
gama_1=doces_dados(linha,31);
gama_2=doces_dados(linha,32);
N=doces_dados(linha,33);
p1=doces_dados(linha,34);
p2=doces_dados(linha,35);
p3=doces_dados(linha,36);
G1=doces_dados(linha,37);
G2=doces_dados(linha,38);
G3=doces_dados(linha,39);
end
```

```
if tipo==4
x_moist=frutos_dados(linha,1);
x_prot=frutos_dados(linha,2);
x_fat=frutos_dados(linha,3);
x_carb=frutos_dados(linha,4);
x_fib=frutos_dados(linha,5);
x_ash=frutos_dados(linha,6);
Temp_ic=frutos_dados(linha,7);
cp_ref=frutos_dados(linha,8);
cp_cong=frutos_dados(linha,9);
calor_lat=frutos_dados(linha,10);
peso=frutos_dados(linha,25);
maior_d=frutos_dados(linha,26);
menor_d=frutos_dados(linha,27);
menor_d2=frutos_dados(linha,28);
dim_caract=frutos_dados(linha,29);
lambda=frutos_dados(linha,30);
gama_1=frutos_dados(linha,31);
gama_2=frutos_dados(linha,32);
N=frutos_dados(linha,33);
p1=frutos_dados(linha,34);
p2=frutos_dados(linha,35);
p3=frutos_dados(linha,36);
G1=frutos_dados(linha,37);
G2=frutos_dados(linha,38);
G3=frutos_dados(linha,39);
end

if tipo==5
x_moist=lact_dados(linha,1);
x_prot=lact_dados(linha,2);
x_fat=lact_dados(linha,3);
x_carb=lact_dados(linha,4);
x_fib=lact_dados(linha,5);
x_ash=lact_dados(linha,6);
Temp_ic=lact_dados(linha,7);
cp_ref=lact_dados(linha,8);
cp_cong=lact_dados(linha,9);
calor_lat=lact_dados(linha,10);
peso=lact_dados(linha,25);
maior_d=lact_dados(linha,26);
menor_d=lact_dados(linha,27);
menor_d2=lact_dados(linha,28);
dim_caract=lact_dados(linha,29);
lambda=lact_dados(linha,30);
gama_1=lact_dados(linha,31);
gama_2=lact_dados(linha,32);
N=lact_dados(linha,33);
p1=lact_dados(linha,34);
p2=lact_dados(linha,35);
p3=lact_dados(linha,36);
G1=lact_dados(linha,37);
G2=lact_dados(linha,38);
G3=lact_dados(linha,39);
end

if tipo==6
x_moist=peixes_dados(linha,1);
x_prot=peixes_dados(linha,2);
x_fat=peixes_dados(linha,3);
x_carb=peixes_dados(linha,4);
```

```
x_fib=peixes_dados(linha,5);
x_ash=peixes_dados(linha,6);
Temp_ic=peixes_dados(linha,7);
cp_ref=peixes_dados(linha,8);
cp_cong=peixes_dados(linha,9);
calor_lat=peixes_dados(linha,10);
peso=peixes_dados(linha,25);
maior_d=peixes_dados(linha,26);
menor_d=peixes_dados(linha,27);
menor_d2=peixes_dados(linha,28);
dim_caract=peixes_dados(linha,29);
lambda=peixes_dados(linha,30);
gama_1=peixes_dados(linha,31);
gama_2=peixes_dados(linha,32);
N=peixes_dados(linha,33);
p1=peixes_dados(linha,34);
p2=peixes_dados(linha,35);
p3=peixes_dados(linha,36);
G1=peixes_dados(linha,37);
G2=peixes_dados(linha,38);
G3=peixes_dados(linha,39);
end

if tipo==7
x_moist=veg_dados(linha,1);
x_prot=veg_dados(linha,2);
x_fat=veg_dados(linha,3);
x_carb=veg_dados(linha,4);
x_fib=veg_dados(linha,5);
x_ash=veg_dados(linha,6);
Temp_ic=veg_dados(linha,7);
cp_ref=veg_dados(linha,8);
cp_cong=veg_dados(linha,9);
calor_lat=veg_dados(linha,10);
peso=veg_dados(linha,25);
maior_d=veg_dados(linha,26);
menor_d=veg_dados(linha,27);
menor_d2=veg_dados(linha,28);
dim_caract=veg_dados(linha,29);
lambda=veg_dados(linha,30);
gama_1=veg_dados(linha,31);
gama_2=veg_dados(linha,32);
N=veg_dados(linha,33);
p1=veg_dados(linha,34);
p2=veg_dados(linha,35);
p3=veg_dados(linha,36);
G1=veg_dados(linha,37);
G2=veg_dados(linha,38);
G3=veg_dados(linha,39);
end

end
```

```
function Q_transp=q_transp(tipo,linha,temp_p,temp_ar)

%Calcula o calor, em W, libertado pela transpiração dos produtos e
%vegetais. Tem como entradas o tipo e linha do produto na respectiva
matriz de propriedades, as temperaturas do produto e do ar envolvente,
em ºC e o local onde o produto se encontra ou vai ser guardado.

pressao_ag=polyfit(evalin('base','pressao_sat(:,1)'),evalin('base','pr
essao_sat(:,2)'),1);
h_sat_ag=polyfit(evalin('base','pressao_sat(:,1)'),evalin('base','pres
sao_sat(:,3)'),1);

pressao_prod=polyval(pressao_ag,temp_p);
pressao_ar=polyval(pressao_ag,temp_ar); %Pela linha de saturação do
ponto de orvalho.
h_sat_pressao_prod=polyval(h_sat_ag,temp_p);

if tipo==4
    k_t=0;
    Area_prod=1;
    if linha==1
        k_t=42e-9;
Area_prod=evalin('base','frutos_dados(1,26)')*evalin('base','frutos_da
dos(1,28)');
    end
    if linha==15
        k_t=81e-9;
Area_prod=evalin('base','frutos_dados(15,26)')*evalin('base','frutos_d
ados(15,28)');
    end
    if linha==16
        k_t=123e-9;
Area_prod=evalin('base','frutos_dados(16,26)')*evalin('base','frutos_d
ados(16,28)');
    end
    if linha==17
        k_t=186e-9;
Area_prod=evalin('base','frutos_dados(17,26)')*evalin('base','frutos_d
ados(17,28)');
    end
    if linha==23
        k_t=117e-9;
Area_prod=evalin('base','frutos_dados(23,26)')*evalin('base','frutos_d
ados(23,28)');
    end
    if linha==24
        k_t=572e-9;
Area_prod=evalin('base','frutos_dados(24,26)')*evalin('base','frutos_d
ados(24,28)');
    end
    if linha==26
        k_t=69e-9;
Area_prod=evalin('base','frutos_dados(26,26)')*evalin('base','frutos_d
ados(26,28)');
    end
    if linha==29
        k_t=136e-9;
Area_prod=evalin('base','frutos_dados(29,26)')*evalin('base','frutos_d
ados(29,28)');
    end
end
```

```
if tipo==7
    k_t=0;
    Area_prod=1;
    if linha==7
        k_t=6150e-9;
Area_prod=evalin('base','veg_dados(7,26)')*evalin('base','veg_dados(7,
28)');
    end
    if linha==8
        k_t=223e-9;
Area_prod=evalin('base','veg_dados(8,26)')*evalin('base','veg_dados(8,
28)');
    end
    if linha==9
        k_t=1207e-9;
Area_prod=evalin('base','veg_dados(9,26)')*evalin('base','veg_dados(9,
28)');
    end
    if linha==12
        k_t=1260e-9;
Area_prod=evalin('base','veg_dados(12,26)')*evalin('base','veg_dados(1
2,28)');
    end
    if linha==20
        k_t=790e-9;
Area_prod=evalin('base','veg_dados(20,26)')*evalin('base','veg_dados(2
0,28)');
    end
    if linha==21
        k_t=7400e-9;
Area_prod=evalin('base','veg_dados(21,26)')*evalin('base','veg_dados(2
1,28)');
    end
    if linha==24
        k_t=60e-9;
Area_prod=evalin('base','veg_dados(24,26)')*evalin('base','veg_dados(2
4,28)');
    end
    if linha==30
        k_t=44e-9;
Area_prod=evalin('base','veg_dados(30,26)')*evalin('base','veg_dados(3
0,28)');
    end
    if linha==37 || linha==38
        k_t=140e-9;
Area_prod=evalin('base','veg_dados(38,26)')*evalin('base','veg_dados(3
8,28)');
    end
end

m_dot=k_t*(pressao_prod-pressao_ar);
Q_transp=m_dot*h_sat_pressao_prod*Area_prod;

end
```

%Procedimento Q_trocado.m;

```
%Este procedimento soma os valores de calor trocados em cada
localização do frigorífico.

i=evalin('base','i'); vol=evalin('base','volume');

if i==1
    importfile_sim('inicio_bc.jou');
else
    importfile_sim('inicio_bc_sim.jou');
end

%Para as bebidas
for m=1:1:16
    assignin('base','m',m);
    massa_1ap_b=evalin('base','beb_dados(m,13)');
    if massa_1ap_b>0
        if i==1
            temp_1ap_b=evalin('base','beb_dados(m,14)');
        else
            temp_1ap_b=evalin('base','r_1ap_b_temp(m,i-1)');
        end
        calor_prod_b(1,m,temp_1ap_b,massa_1ap_b,2);
    end
    massa_2ap_b=evalin('base','beb_dados(m,15)');
    if massa_2ap_b>0
        if i==1
            temp_2ap_b=evalin('base','beb_dados(m,16)');
        else
            temp_2ap_b=evalin('base','r_2ap_b_temp(m,i-1)');
        end
        calor_prod_b(1,m,temp_2ap_b,massa_2ap_b,3);
    end
    massa_3ap_b=evalin('base','beb_dados(m,17)');
    if massa_3ap_b>0
        if i==1
            temp_3ap_b=evalin('base','beb_dados(m,18)');
        else
            temp_3ap_b=evalin('base','r_3ap_b_temp(m,i-1)');
        end
        calor_prod_b(1,m,temp_3ap_b,massa_3ap_b,4);
    end
    massa_4ap_b=evalin('base','beb_dados(m,19)');
    if massa_4ap_b>0
        if i==1
            temp_4ap_b=evalin('base','beb_dados(m,20)');
        else
            temp_4ap_b=evalin('base','r_4ap_b_temp(m,i-1)');
        end
        calor_prod_b(1,m,temp_4ap_b,massa_4ap_b,5);
    end
    massa_gav_b=evalin('base','beb_dados(m,21)');
    if massa_gav_b>0
        if i==1
            temp_gav_b=evalin('base','beb_dados(m,22)');
        else
            temp_gav_b=evalin('base','r_g_b_temp(m,i-1)');
        end
        calor_prod_b(1,m,temp_gav_b,massa_gav_b,6);
    end
```

```
    massa_p_b=evalin('base','beb_dados(m,11)');
    if massa_p_b>0
        if i==1
            temp_p_b=evalin('base','beb_dados(m,12)');
        else
            temp_p_b=evalin('base','r_p_b_temp(m,i-1)');
        end
        calor_prod_b(1,m,temp_p_b,massa_p_b,7);
    end
    massa_c_b=evalin('base','beb_dados(m,23)');
    if massa_c_b>0
        if i==1
            temp_c_b=evalin('base','beb_dados(m,24)');
        else
            temp_c_b=evalin('base','c_b_temp(m,i-1)');
        end
        calor_prod_b(1,m,temp_c_b,massa_c_b,1);
    end
end
%Para as carnes.
for m=1:1:36
    massa_1ap_c=evalin('base','carne_dados(m,13)');
    if massa_1ap_c>0
        if i==1
            temp_1ap_c=evalin('base','carne_dados(m,14)');
        else
            temp_1ap_c=evalin('base','r_1ap_c_temp(m,i-1)');
        end
        calor_prod_c(2,m,temp_1ap_c,massa_1ap_c,2);
    end
    massa_2ap_c=evalin('base','carne_dados(m,15)');
    if massa_2ap_c>0
        if i==1
            temp_2ap_c=evalin('base','carne_dados(m,16)');
        else
            temp_2ap_c=evalin('base','r_2ap_c_temp(m,i-1)');
        end
        calor_prod_c(2,m,temp_2ap_c,massa_2ap_c,3);
    end
    massa_3ap_c=evalin('base','carne_dados(m,17)');
    if massa_3ap_c>0
        if i==1
            temp_3ap_c=evalin('base','carne_dados(m,18)');
        else
            temp_3ap_c=evalin('base','r_1ap_c_temp(m,i-1)');
        end
        calor_prod_c(2,m,temp_3ap_c,massa_3ap_c,4);
    end
    massa_4ap_c=evalin('base','carne_dados(m,19)');
    if massa_4ap_c>0
        if i==1
            temp_4ap_c=evalin('base','carne_dados(m,20)');
        else
            temp_4ap_c=evalin('base','r_4ap_c_temp(m,i-1)');
        end
        calor_prod_c(2,m,temp_4ap_c,massa_4ap_c,5);
    end
    massa_c_c=evalin('base','carne_dados(m,23)');
    if massa_c_c>0
        if i==1
            temp_c_c=evalin('base','carne_dados(m,24)');
```

```
        else
            temp_c_c=evalin('base','c_c_temp(m,i-1)');
        end
        calor_prod_c(2,m,temp_c_c,massa_c_c,1);
    end
end
massa_p_c=evalin('base','carne_dados(30,11)');
if massa_p_c>0
    if i==1
        temp_p_c=evalin('base','carne_dados(30,12)');
    else
        temp_p_c=evalin('base','r_p_ovos_temp(1,i-1)');
    end
    calor_prod_c(2,30,temp_p_c,massa_p_c,7);
end
%Para os doces.
for m=1:1:14
    assignin('base','m',m);
    massa_1ap_d=evalin('base','doces_dados(m,13)');
    if massa_1ap_d>0
        if i==1
            temp_1ap_d=evalin('base','doces_dados(m,14)');
        else
            temp_1ap_d=evalin('base','r_1ap_d_temp(m,i-1)');
        end
        calor_prod_d(3,m,temp_1ap_d,massa_1ap_d,2);
    end
    massa_2ap_d=evalin('base','doces_dados(m,15)');
    if massa_2ap_d>0
        if i==1
            temp_2ap_d=evalin('base','doces_dados(m,16)');
        else
            temp_2ap_d=evalin('base','r_2ap_d_temp(m,i-1)');
        end
        calor_prod_d(3,m,temp_2ap_d,massa_2ap_d,3);
    end
    massa_3ap_d=evalin('base','doces_dados(m,17)');
    if massa_3ap_d>0
        if i==1
            temp_3ap_d=evalin('base','doces_dados(m,18)');
        else
            temp_3ap_d=evalin('base','r_3ap_d_temp(m,i-1)');
        end
        calor_prod_d(3,m,temp_3ap_d,massa_3ap_d,4);
    end
    massa_4ap_d=evalin('base','doces_dados(m,19)');
    if massa_4ap_d>0
        if i==1
            temp_4ap_d=evalin('base','doces_dados(m,20)');
        else
            temp_4ap_d=evalin('base','r_4ap_d_temp(m,i-1)');
        end
        calor_prod_d(3,m,temp_4ap_d,massa_4ap_d,5);
    end
    massa_c_d=evalin('base','doces_dados(m,23)');
    if massa_c_d>0
        if i==1
            temp_c_d=evalin('base','doces_dados(m,24)');
        else
            temp_c_d=evalin('base','c_d_temp(m,i-1)');
        end
```

```
        calor_prod_d(3,m,temp_c_d,massa_c_d,1);
    end
end
%Para os frutos.
for m=1:1:38
    massa_1ap_f=evalin('base','frutos_dados(m,13)');
    if massa_1ap_f>0
        if i==1
            temp_1ap_f=evalin('base','frutos_dados(m,14)');
        else
            temp_1ap_f=evalin('base','r_1ap_f_temp(m,i-1)');
        end
        calor_prod_f(4,m,temp_1ap_f,massa_1ap_f,2);

calor_transp_1ap_f=q_transp(4,m,temp_1ap_f,evalin('base','Temp_1ap'));

evalin('base',sprintf('r_1ap_f_transp(m,i)=%u',calor_transp_1ap_f));
    end
    massa_2ap_f=evalin('base','frutos_dados(m,15)');
    if massa_2ap_f>0
        if i==1
            temp_2ap_f=evalin('base','frutos_dados(m,16)');
        else
            temp_2ap_f=evalin('base','r_2ap_f_temp(m,i-1)');
        end
        calor_prod_f(4,m,temp_2ap_f,massa_2ap_f,3);

calor_transp_2ap_f=q_transp(4,m,temp_2ap_f,evalin('base','Temp_2ap'));

evalin('base',sprintf('r_2ap_f_transp(m,i)=%u',calor_transp_2ap_f));
    end
    massa_3ap_f=evalin('base','frutos_dados(m,17)');
    if massa_3ap_f>0
        if i==1
            temp_3ap_f=evalin('base','frutos_dados(m,18)');
        else
            temp_3ap_f=evalin('base','r_3ap_f_temp(m,i-1)');
        end
        calor_prod_f(4,m,temp_3ap_f,massa_3ap_f,4);
calor_transp_3ap_f=q_transp(4,m,temp_3ap_f,evalin('base','Temp_3ap'));
evalin('base',sprintf('r_3ap_f_transp(m,i)=%u',calor_transp_3ap_f));
    end
    massa_4ap_f=evalin('base','frutos_dados(m,19)');
    if massa_4ap_f>0
        if i==1
            temp_4ap_f=evalin('base','frutos_dados(m,20)');
        else
            temp_4ap_f=evalin('base','r_4ap_f_temp(m,i-1)');
        end
        calor_prod_f(4,m,temp_4ap_f,massa_4ap_f,5);
calor_transp_4ap_f=q_transp(4,m,temp_4ap_f,evalin('base','Temp_4ap'));
evalin('base',sprintf('r_4ap_f_transp(m,i)=%u',calor_transp_4ap_f));
    end
    massa_gav_f=evalin('base','frutos_dados(m,21)');
    if massa_gav_f>0
        if i==1
            temp_gav_f=evalin('base','frutos_dados(m,22)');
        else
            temp_gav_f=evalin('base','r_g_f_temp(m,i-1)');
        end
        calor_prod_f(4,m,temp_gav_f,massa_gav_f,6);
```

```
calor_transp_gav_f=q_transp(4,m,temp_gav_f,evalin('base','Temp_gav'));
evalin('base',sprintf('r_g_f_transp(m,i)=%u',calor_transp_gav_f));
    end
    massa_c_f=evalin('base','frutos_dados(m,23)');
    if massa_c_f>0
        if i==1
            temp_c_f=evalin('base','frutos_dados(m,24)');
        else
            temp_c_f=evalin('base','c_f_temp(m,i-1)');
        end
        calor_prod_f(4,m,temp_c_f,massa_c_f,1);
calor_transp_c_f=q_transp(4,m,temp_c_f,evalin('base','Temp_c'));
        evalin('base',sprintf('c_f_transp(m,i)=%u',calor_transp_c_f));
    end
end
%Para os lacticinios.
for m=1:1:27
    massa_1ap_l=evalin('base','lact_dados(m,13)');
    if massa_1ap_l>0
        if i==1
            temp_1ap_l=evalin('base','lact_dados(m,14)');
        else
            temp_1ap_l=evalin('base','r_1ap_l_temp(m,i-1)');
        end
        calor_prod_l(5,m,temp_1ap_l,massa_1ap_l,2);
    end
    massa_2ap_l=evalin('base','lact_dados(m,15)');
    if massa_2ap_l>0
        if i==1
            temp_2ap_l=evalin('base','lact_dados(m,16)');
        else
            temp_2ap_l=evalin('base','r_2ap_l_temp(m,i-1)');
        end
        calor_prod_l(5,m,temp_2ap_l,massa_2ap_l,3);
    end
    massa_3ap_l=evalin('base','lact_dados(m,17)');
    if massa_3ap_l>0
        if i==1
            temp_3ap_l=evalin('base','lact_dados(m,18)');
        else
            temp_3ap_l=evalin('base','r_3ap_l_temp(m,i-1)');
        end
        calor_prod_l(5,m,temp_3ap_l,massa_3ap_l,4);
    end
    massa_4ap_l=evalin('base','lact_dados(m,19)');
    if massa_4ap_l>0
        if i==1
            temp_4ap_l=evalin('base','lact_dados(m,20)');
        else
            temp_4ap_l=evalin('base','r_4ap_l_temp(m,i-1)');
        end
        calor_prod_l(5,m,temp_4ap_l,massa_4ap_l,5);
    end
    massa_p_l=evalin('base','lact_dados(m,11)');
    if massa_p_l>0
        if i==1
            temp_p_l=evalin('base','lact_dados(m,12)');
        else
            temp_p_l=evalin('base','r_p_l_temp(m,i-1)');
        end
        calor_prod_l(5,m,temp_p_l,massa_p_l,7);
```

```
    end
    massa_c_l=evalin('base','lact_dados(m,23)');
    if massa_c_l>0
        if i==1
            temp_c_l=evalin('base','lact_dados(m,24)');
        else
            temp_c_l=evalin('base','c_l_temp(m,i-1)');
        end
        calor_prod_l(5,m,temp_c_l,massa_c_l,1);
    end
end
%Para os peixes.
for m=1:1:15
    massa_1ap_p=evalin('base','peixes_dados(m,13)');
    if massa_1ap_p>0
        if i==1
            temp_1ap_p=evalin('base','peixes_dados(m,14)');
        else
            temp_1ap_p=evalin('base','r_1ap_p_temp(m,i-1)');
        end
        calor_prod_p(6,m,temp_1ap_p,massa_1ap_p,2);
    end
    massa_2ap_p=evalin('base','peixes_dados(m,15)');
    if massa_2ap_p>0
        if i==1
            temp_2ap_p=evalin('base','peixes_dados(m,16)');
        else
            temp_2ap_p=evalin('base','r_2ap_p_temp(m,i-1)');
        end
        calor_prod_p(6,m,temp_2ap_p,massa_2ap_p,3);
    end
    massa_3ap_p=evalin('base','peixes_dados(m,17)');
    if massa_3ap_p>0
        calor_prod_p(6,m,temp_3ap_p,massa_3ap_p,4);
        if i==1
            temp_3ap_p=evalin('base','peixes_dados(m,18)');
        else
            temp_3ap_p=evalin('base','r_3ap_p_temp(m,i-1)');
        end
    end
    massa_4ap_p=evalin('base','peixes_dados(m,19)');
    if massa_4ap_p>0
        if i==1
            temp_4ap_p=evalin('base','peixes_dados(m,20)');
        else
            temp_4ap_p=evalin('base','r_4ap_p_temp(m,i-1)');
        end
        calor_prod_p(6,m,temp_4ap_p,massa_4ap_p,5);
    end
    massa_c_p=evalin('base','peixes_dados(m,23)');
    if massa_c_p>0
        if i==1
            temp_c_p=evalin('base','peixes_dados(m,24)');
        else
            temp_c_p=evalin('base','c_p_temp(m,i-1)');
        end
        calor_prod_p(6,m,temp_c_p,massa_c_p,1);
    end
end
%Para os vegetais.
for m=1:1:42
```

```
    massa_1ap_v=evalin('base','veg_dados(m,13)');
    if massa_1ap_v>0
        if i==1
            temp_1ap_v=evalin('base','veg_dados(m,14)');
        else
            temp_1ap_v=evalin('base','r_1ap_v_temp(m,i-1)');
        end
        calor_prod_v(7,m,temp_1ap_v,massa_1ap_v,2);
calor_transp_1ap_v=q_transp(7,m,temp_1ap_v,evalin('base','Temp_1ap'));
evalin('base',sprintf('r_1ap_v_transp(m,i)=%u',calor_transp_1ap_v));
    end
    massa_2ap_v=evalin('base','veg_dados(m,15)');
    if massa_2ap_v>0
        if i==1
            temp_2ap_v=evalin('base','veg_dados(m,16)');
        else
            temp_2ap_v=evalin('base','r_2ap_v_temp(m,i-1)');
        end
        calor_prod_v(7,m,temp_2ap_v,massa_2ap_v,3);
calor_transp_2ap_v=q_transp(7,m,temp_2ap_v,evalin('base','Temp_2ap'));
evalin('base',sprintf('r_2ap_v_transp(m,i)=%u',calor_transp_2ap_v));
    end
    massa_3ap_v=evalin('base','veg_dados(m,17)');
    if massa_3ap_v>0
        if i==1
            temp_3ap_v=evalin('base','veg_dados(m,18)');
        else
            temp_3ap_v=evalin('base','r_3ap_v_temp(m,i-1)');
        end
        calor_prod_v(7,m,temp_3ap_v,massa_3ap_v,4);
calor_transp_3ap_v=q_transp(7,m,temp_3ap_v,evalin('base','Temp_3ap'));
evalin('base',sprintf('r_3ap_v_transp(m,i)=%u',calor_transp_3ap_v));
    end
    massa_4ap_v=evalin('base','veg_dados(m,19)');
    if massa_4ap_v>0
        if i==1
            temp_4ap_v=evalin('base','veg_dados(m,20)');
        else
            temp_4ap_v=evalin('base','r_4ap_v_temp(m,i-1)');
        end
        calor_prod_v(7,m,temp_4ap_v,massa_4ap_v,5);
calor_transp_4ap_v=q_transp(7,m,temp_4ap_v,evalin('base','Temp_4ap'));
evalin('base',sprintf('r_4ap_v_transp(m,i)=%u',calor_transp_4ap_v));
    end
    massa_gav_v=evalin('base','veg_dados(m,21)');
    if massa_gav_v>0
        if i==1
            temp_gav_v=evalin('base','veg_dados(m,22)');
        else
            temp_gav_v=evalin('base','r_g_v_temp(m,i-1)');
        end
        calor_prod_v(7,m,temp_gav_v,massa_gav_v,6);
calor_transp_gav_v=q_transp(7,m,temp_gav_v,evalin('base','Temp_gav'));
evalin('base',sprintf('r_g_v_transp(m,i)=%u',calor_transp_gav_v));
    end
    massa_p_v=evalin('base','veg_dados(m,11)');
    massa_c_v=evalin('base','veg_dados(m,23)');
    if massa_c_v>0
        if i==1
            temp_c_v=evalin('base','veg_dados(m,24)');
        else
```

```
            temp_c_v=evalin('base','c_v_temp(m,i-1)');
        end
        calor_prod_v(7,m,temp_c_v,massa_c_v,1);
calor_transp_c_v=q_transp(7,m,temp_c_v,evalin('base','Temp_c'));
        evalin('base',sprintf('c_v_transp(m,i)=%u',calor_transp_c_v));
    end
end


%1a prateleira;
calor_1pr=sum(evalin('base','r_1ap_b(:,i)'))+sum(evalin('base','r_1ap_
c(:,i)'))+sum(evalin('base','r_1ap_d(:,i)'))+sum(evalin('base','r_1ap_
f(:,i)'))+sum(evalin('base','r_1ap_f_resp(:,i)'))+sum(evalin('base','r
_1ap_f_transp(:,i)'))+sum(evalin('base','r_1ap_l(:,i)'))+sum(evalin('b
ase','r_1ap_p(:,i)'))+sum(evalin('base','r_1ap_v(:,i)'))+sum(evalin('b
ase','r_1ap_v_resp(:,i)'))+sum(evalin('base','r_1ap_v_transp(:,i)'));
evalin('base',sprintf('calor_1p(1,i)=%u',i*2));
evalin('base',sprintf('calor_1p(2,i)=%u',calor_1pr));
if vol==200
    calor1_num=(calor_1pr/0.367)/6;
    calor1=num2str(calor1_num);
end
if i==1
inicio_bc{15,1}=calor1;inicio_bc{33,1}=calor1;inicio_bc{51,1}=calor1;
inicio_bc{69,1}=calor1;inicio_bc{87,1}=calor1;inicio_bc{105,1}=calor1;
else
inicio_bc_sim{15,1}=calor1;inicio_bc_sim{33,1}=calor1;inicio_bc_sim{51
,1}=calor1;inicio_bc_sim{69,1}=calor1;inicio_bc_sim{87,1}=calor1;
inicio_bc_sim{105,1}=calor1;
end

%2a prateleira;
calor_2pr=sum(evalin('base','r_2ap_b(:,i)'))+sum(evalin('base','r_2ap_
c(:,i)'))+sum(evalin('base','r_2ap_d(:,i)'))+sum(evalin('base','r_2ap_
f(:,i)'))+sum(evalin('base','r_2ap_f_resp(:,i)'))+sum(evalin('base','r
_2ap_f_transp(:,i)'))+sum(evalin('base','r_2ap_l(:,i)'))+sum(evalin('b
ase','r_2ap_p(:,i)'))+sum(evalin('base','r_2ap_v(:,i)'))+sum(evalin('b
ase','r_2ap_v_resp(:,i)'))+sum(evalin('base','r_2ap_v_transp(:,i)'));
evalin('base',sprintf('calor_2p(1,i)=%u',i*2));
evalin('base',sprintf('calor_2p(2,i)=%u',calor_2pr));
if vol==200
    calor2_num=(calor_2pr/0.367)/6;calor2=num2str(calor2_num);
end
if i==1
inicio_bc{123,1}=calor2;inicio_bc{141,1}=calor2;inicio_bc{159,1}=calor
2;inicio_bc{177,1}=calor2;inicio_bc{195,1}=calor2;inicio_bc{213,1}=cal
or2;
else
inicio_bc_sim{123,1}=calor2;inicio_bc_sim{141,1}=calor2;inicio_bc_sim{
159,1}=calor2;inicio_bc_sim{177,1}=calor2;inicio_bc_sim{195,1}=calor2;
inicio_bc_sim{213,1}=calor2;
end

%3a prateleira;
calor_3pr=sum(evalin('base','r_3ap_b(:,i)'))+sum(evalin('base','r_3ap_
c(:,i)'))+sum(evalin('base','r_3ap_d(:,i)'))+sum(evalin('base','r_3ap_
f(:,i)'))+sum(evalin('base','r_3ap_f_resp(:,i)'))+sum(evalin('base','r
_3ap_f_transp(:,i)'))+sum(evalin('base','r_3ap_l(:,i)'))+sum(evalin('b
ase','r_3ap_p(:,i)'))+sum(evalin('base','r_3ap_v(:,i)'))+sum(evalin('b
ase','r_3ap_v_resp(:,i)'))+sum(evalin('base','r_3ap_v_transp(:,i)'));
evalin('base',sprintf('calor_3p(1,i)=%u',i*2));
```

```
evalin('base',sprintf('calor_3p(2,i)=%u',calor_3pr));
if vol==200
    calor3_num=(calor_3pr/0.367)/6;calor3=num2str(calor3_num);
end
if i==1
inicio_bc{231,1}=calor3;inicio_bc{249,1}=calor3;inicio_bc{267,1}=calor
3;inicio_bc{285,1}=calor3;inicio_bc{303,1}=calor3;inicio_bc{321,1}=cal
or3;
else
inicio_bc_sim{231,1}=calor3;inicio_bc_sim{249,1}=calor3;inicio_bc_sim{
267,1}=calor3;inicio_bc_sim{285,1}=calor3;inicio_bc_sim{303,1}=calor3;
inicio_bc_sim{321,1}=calor3;
end

%4a prateleira;
calor_4pr=sum(evalin('base','r_4ap_b(:,i)'))+sum(evalin('base','r_4ap_
c(:,i)'))+sum(evalin('base','r_4ap_d(:,i)'))+sum(evalin('base','r_4ap_
f(:,i)'))+sum(evalin('base','r_4ap_f_resp(:,i)'))+sum(evalin('base','r
_4ap_f_transp(:,i)'))+sum(evalin('base','r_4ap_l(:,i)'))+sum(evalin('b
ase','r_4ap_p(:,i)'))+sum(evalin('base','r_4ap_v(:,i)'))+sum(evalin('b
ase','r_4ap_v_resp(:,i)'))+sum(evalin('base','r_4ap_v_transp(:,i)'));
evalin('base',sprintf('calor_4p(1,i)=%u',i*2));
evalin('base',sprintf('calor_4p(2,i)=%u',calor_4pr));
if vol==200
    calor4_num=(calor_4pr/0.367)/6;calor4=num2str(calor4_num);
end
if i==1
inicio_bc{339,1}=calor4;inicio_bc{357,1}=calor4;inicio_bc{375,1}=calor
4;inicio_bc{393,1}=calor4;inicio_bc{411,1}=calor4;inicio_bc{429,1}=cal
or4;
else
inicio_bc_sim{339,1}=calor4;inicio_bc_sim{357,1}=calor4;inicio_bc_sim{
375,1}=calor4;inicio_bc_sim{393,1}=calor4;inicio_bc_sim{411,1}=calor4;
inicio_bc_sim{429,1}=calor4;
end

%gavetas;
calor_ga=sum(evalin('base','r_g_f(:,i)'))+sum(evalin('base','r_g_f_res
p(:,i)'))+sum(evalin('base','r_g_f_transp(:,i)'))+sum(evalin('base','r
_g_v(:,i)'))+sum(evalin('base','r_g_v_resp(:,i)'))+sum(evalin('base','
r_g_v_transp(:,i)'));
evalin('base',sprintf('calor_g(1,i)=%u',i*2));
evalin('base',sprintf('calor_g(2,i)=%u',calor_ga));
if vol==200
    calorgav_num=(calor_ga/0.367)/6;calorgav=num2str(calorgav_num);
end
if i==1
inicio_bc{645,1}=calorgav;inicio_bc{663,1}=calorgav;inicio_bc{681,1}=c
alorgav;inicio_bc{699,1}=calorgav;inicio_bc{717,1}=calorgav;
else
inicio_bc_sim{645,1}=calorgav;inicio_bc_sim{663,1}=calorgav;inicio_bc_
sim{681,1}=calorgav;inicio_bc_sim{699,1}=calorgav;inicio_bc_sim{717,1}
=calorgav;
end

%congelador;
calor_co=sum(evalin('base','c_b(:,i)'))+sum(evalin('base','c_c(:,i)'))
+sum(evalin('base','c_d(:,i)'))+sum(evalin('base','c_f(:,i)'))+sum(eva
lin('base','c_f_resp(:,i)'))+sum(evalin('base','c_f_transp(:,i)'))+sum
(evalin('base','c_l(:,i)'))+sum(evalin('base','c_p(:,i)'))+sum(evalin(
```

```
'base','c_v(:,i)'))+sum(evalin('base','c_f_resp(:,i)'))+sum(evalin('ba
se','c_v_transp(:,i)'));
evalin('base',sprintf('calor_c(1,i)=%u',i*2));
evalin('base',sprintf('calor_c(2,i)=%u',calor_co));
calor_infiltr_c=infiltr(2,evalin('base','novo_velho'),1,evalin('base',
'volume'),evalin('base','Temp'),evalin('base','Temp_r'),evalin('base',
'Temp_c'));
[q_c_e,q_c_d,q_c_f]=calor_paredes_r(evalin('base','Temp'),evalin('base
','Temp_r'),evalin('base','Temp_c'),2,0.4,0.5,0.6,0.001,0.055,0.001);
calor_envolv_c=(q_c_e+q_c_d+q_c_f)/3;
calor_renov_ar_c=renov_ar(1,0.2,0.4);
evalin('base',sprintf('c_infilt(1,i)=%u',calor_infiltr_c));
evalin('base',sprintf('c_envolv(1,i)=%u',calor_envolv_c+calor_renov_ar
_c));
calor_envolv_cong_num=calor_infiltr_c+calor_envolv_c+calor_renov_ar_c;
calor_envolv_cong=num2str(calor_envolv_cong_num);
if vol==200
    calorc_num=(calor_co/0.475)/6;calorc=num2str(calorc_num);
end
if i==1
inicio_bc{447,1}=calorc;inicio_bc{465,1}=calorc;inicio_bc{483,1}=calor
c;inicio_bc{501,1}=calorc;inicio_bc{519,1}=calorc;inicio_bc{537,1}=cal
orc;inicio_bc{857,1}=calor_envolv_cong;
else
inicio_bc_sim{447,1}=calorc;inicio_bc_sim{465,1}=calorc;inicio_bc_sim{
483,1}=calorc;inicio_bc_sim{501,1}=calorc;inicio_bc_sim{519,1}=calorc;
inicio_bc_sim{537,1}=calorc;inicio_bc_sim{857,1}=calor_envolv_cong;
end

%porta;
calor_po=sum(evalin('base','r_p_b(:,i)'))+sum(evalin('base','r_p_l(:,i
)')); calor_ovos=evalin('base','r_p_ovos(1,i)');
evalin('base',sprintf('calor_p(1,i)=%u',i*2));
evalin('base',sprintf('calor_p(2,i)=%u',calor_po));
if vol==200
    calorp_num=(calor_po/0.385)/6;calorp=num2str(calorp_num);
    calorovos_num=(calor_ovos/0.133)/6;
    calorovos=num2str(calorovos_num);
end
if i==1
inicio_bc{555,1}=calorp;inicio_bc{573,1}=calorp;inicio_bc{591,1}=calor
p;inicio_bc{609,1}=calorp;inicio_bc{627,1}=calorp;inicio_bc{735,1}=cal
orovos;inicio_bc{753,1}=calorovos;inicio_bc{771,1}=calorovos;inicio_bc
{789,1}=calorovos;inicio_bc{807,1}=calorovos;
else
inicio_bc_sim{555,1}=calorp;inicio_bc_sim{573,1}=calorp;inicio_bc_sim{
591,1}=calorp;inicio_bc_sim{609,1}=calorp;inicio_bc_sim{627,1}=calorp;
inicio_bc_sim{735,1}=calorovos;inicio_bc_sim{753,1}=calorovos;inicio_b
c_sim{771,1}=calorovos;inicio_bc_sim{789,1}=calorovos;inicio_bc_sim{80
7,1}=calorovos;
end

%refrigerador completo;
calor_infiltr_r=infiltr(30,evalin('base','novo_velho'),2,evalin('base'
,'volume'),evalin('base','Temp'),evalin('base','Temp_r'),evalin('base'
,'Temp_c'));
[q_r_e,q_r_d,q_r_f]=calor_paredes_r(evalin('base','Temp'),evalin('base
','Temp_r'),evalin('base','Temp_c'),1,1.3,0.5,0.6,0.001,0.055,0.001);
calor_envolv_r=(q_r_e+q_r_d+q_r_f)/3;
calor_renov_ar_r=renov_ar(2,0.65,1.3);
calor_envolv_refr_num=calor_infiltr_r+calor_envolv_r+calor_renov_ar_r;
```

```
calor_envolv_refr=num2str(calor_envolv_refr_num);
calor_prod_total=sum(evalin('base','calor_1p(1,1:i)')+evalin('base','c
alor_2p(1,1:i)')+evalin('base','calor_3p(1,1:i)')+evalin('base','calor
_4p(1,1:i)')+evalin('base','calor_g(1,1:i)')+evalin('base','calor_p(1,
1:i)')+evalin('base','calor_c(1,1:i)'));
calor_prod_resp_total=sum(evalin('base','r_1ap_f_resp(:,i)')+evalin('b
ase','r_2ap_f_resp(:,i)')+evalin('base','r_3ap_f_resp(:,i)')+evalin('b
ase','r_4ap_f_resp(:,i)')+evalin('base','r_g_f_resp(:,i)')+evalin('bas
e','c_f_resp(:,i)')+evalin('base','r_1ap_v_resp(:,i)')+evalin('base','
r_2ap_v_resp(:,i)')+evalin('base','r_3ap_v_resp(:,i)')+evalin('base','
r_4ap_v_resp(:,i)')+evalin('base','r_g_v_resp(:,i)')+evalin('base','c_
v_resp(:,i)'));
calor_prod_transp_total=sum(evalin('base','r_1ap_f_transp(:,i)'))+sum(
evalin('base','r_2ap_f_transp(:,i)'))+sum(evalin('base','r_3ap_f_trans
p(:,i)'))+sum(evalin('base','r_4ap_f_transp(:,i)'))+sum(evalin('base',
'r_g_f_transp(:,i)'))+sum(evalin('base','c_f_transp(:,i)'))+sum(evalin
('base','r_1ap_v_transp(:,i)'))+sum(evalin('base','r_2ap_v_transp(:,i)
'))+sum(evalin('base','r_3ap_v_transp(:,i)'))+sum(evalin('base','r_4ap
_v_transp(:,i)'))+sum(evalin('base','r_g_v_transp(:,i)'))+sum(evalin('
base','c_v_transp(:,i)'));
evalin('base',sprintf('calor_prod_t(1,i)=%u',calor_prod_total));
evalin('base',sprintf('calor_prod_resp(1,i)=%u',calor_prod_resp_total)
);
evalin('base',sprintf('calor_prod_transp(1,i)=%u',calor_prod_transp_to
tal));
evalin('base',sprintf('r_infilt(1,i)=%u',calor_infiltr_r));
evalin('base',sprintf('r_envolv(1,i)=%u',calor_envolv_r+calor_renov_ar
_r));
evalin('base',sprintf('temp_frig(1,i)=%u',i*2));
evalin('base',sprintf('temp_frig(2,i)=%u',evalin('base','Temp_1ap')));
evalin('base',sprintf('temp_frig(3,i)=%u',evalin('base','Temp_2ap')));
evalin('base',sprintf('temp_frig(4,i)=%u',evalin('base','Temp_3ap')));
evalin('base',sprintf('temp_frig(5,i)=%u',evalin('base','Temp_4ap')));
evalin('base',sprintf('temp_frig(6,i)=%u',evalin('base','Temp_gav')));
evalin('base',sprintf('temp_frig(7,i)=%u',evalin('base','Temp_porta'))
);
evalin('base',sprintf('temp_frig(8,i)=%u',evalin('base','Temp_r')));
evalin('base',sprintf('temp_frig(9,i)=%u',evalin('base','Temp_c')));

if i==1
    inicio_bc{875,1}=calor_envolv_refr;
else
    inicio_bc_sim{875,1}=calor_envolv_refr;
end
l=evalin('base','i');
if l==1
    k=num2str(l);inicio_bc{1049,1}=k;
else
    k=num2str(l);m=num2str(l-1);inicio_bc_sim{3,1}=m;
    inicio_bc_sim{1049,1}=k;
end
evalin('base','clc');
fid=fopen('inicio_bc_sim.jou','wt');
if l==1
    evalin('base','fprintf(fid,''%s\n'',inicio_bc{:})');
else
    evalin('base','fprintf(fid,''%s\n'',inicio_bc_sim{:})');
end
evalin('base','fclose(fid)');

clear inicio_bc inicio_bc_sim;
```

```
function Q_renov=renov_ar(cong_refr,Area_porta,alt_porta)

%calcula o valor da potência que entra no frigorífico, em W, devida à
%abertura das portas. Tem como entradas a indicação de se tratar do
cálculo para o congelador ou para o refrigerador, a área da porta do
local respectivo, em m2, e a altura da mesma porta, em m.

if cong_refr==1
    [ro_frig,h_ar_frig]=propriedad_ar(evalin('base','Temp_c'));
else
    [ro_frig,h_ar_frig]=propriedad_ar(evalin('base','Temp_r'));
end

[ro_inf,h_ar_inf]=propriedad_ar(evalin('base','Temp'));Efic=0.95;
P=evalin('base','n_vezes_abert');teta_p=1;
teta_o=(evalin('base','tempo_medio_abert'))/60;
teta_d=(evalin('base','periodo_simul'));Df=0.8;

Fm=(2/(1+(ro_frig/ro_inf)^(1/3)))^1.5;
Q_renov_diario=0.221*Area_porta*(h_ar_inf-h_ar_frig)*ro_frig*(1-
ro_inf/ro_frig)^0.5*(9.81*alt_porta)^0.5*Fm;

Dt=(P*teta_p+60*teta_o)/(3600*teta_d);
Q_renov=((Q_renov_diario*Dt*Df*(1-Efic))*1e3)/720; %720 blocos de 2
minutos em cada dia;

end

function[t_c]=tempo_c(tipo,linha,temp_p,temp_c)
%Determina o tempo de congelamento de um produto, em s, desde a sua
%temperatura inicial até à temperatura da sua envolvente. Tem como
entradas o tipo e a linha do produto na respectiva matriz de
propriedades e as temperaturas do produto e do congelador, em ºC.

if temp_p==temp_c
    t_c=0;
else

[xm,xp,xfat,xc,xfib,xa,Tempc,cpr,cpc,calor_l,kg,maiord,menord,menor2d,
dimcar,lambda,gama1,gama2,N,p1,p2,p3,G1,G2,G3]=propriedades(tipo,linha
); beta1=menor2d/menord; beta2=maiord/menord;

if tipo==1
alfa=7.7;
end
if tipo==2
alfa=10.0;
end
if tipo==3
alfa=23.8;
end
if tipo==4
alfa=23.8;
end
if tipo==5
alfa=8;
end
if tipo==6
alfa=34.3;
```

```
end
if tipo==7
alfa=14.0;
end

temp_ff=-40;k_t=condtermica(tipo,linha,temp_ff);
ro_tp=densidade(tipo,linha,temp_p);ro_tc=densidade(tipo,linha,temp_ff)
;entalpia_tp=entalpia_r(tipo,linha,temp_p,temp_c);
[entalpia_ic,entalpia_tc]=entalpia_c(tipo,linha,temp_c);
cp_c=(caloresp_c(tipo,linha,temp_ff))*1e+03;
cp_r=(caloresp_r(tipo,linha))*1e+03;

if temp_p<Tempc
entalpia_inicial=entalpia_tc*1e+03;
end
if temp_p==Tempc
entalpia_inicial=entalpia_ic*1e+03;
end
if temp_p>Tempc
entalpia_inicial=entalpia_tp*1e+03;
end

%Para congelamento, a dimcar é 2 vezes a distância mais curta do
centro térmico à superfície do produto.
biot=alfa*dimcar*2/k_t;
deltaH18=ro_tp*entalpia_inicial-ro_tc*(entalpia_tc*1e+03);
pk=cp_r*ro_tp*(temp_p-Tempc)/deltaH18;
ste=cp_c*ro_tc*(Tempc-temp_c)/deltaH18;
deltaT=(Tempc-temp_c)+(((temp_p-Tempc)^2*(cp_r*ro_tp)/2)-((Tempc-
(temp_c))^2*(cp_c*ro_tc)/2))/deltaH18;

U=deltaT/(Tempc-temp_c);
P=0.7306-1.083*pk+ste*(15.4*U-15.43+0.01329*ste/biot);
R=0.2079-0.2656*U*ste;
t_c_p=(deltaH18/deltaT)*(P*dimcar*2/alfa+R*(2*dimcar)^2/k_t);

E1=((2.32/(beta1^1.77))/((biot^1.34)+2.32/(beta1^1.77)))*1/beta1+0.73/
(biot^2.5)*(1-((2.32/(beta1^1.77))/((biot^1.34)+2.32/(beta1^1.77))));

E2=((2.32/(beta2^1.77))/((biot^1.34)+2.32/(beta2^1.77)))*1/beta2+0.73/
(biot^2.5)*(1-((2.32/(beta2^1.77))/((biot^1.34)+2.32/(beta2^1.77))));
    E=G1+G2*E1+G3*E2;

    t_c=t_c_p/E;
end
end
```

```
function[t_r]=tempo_r(tipo,linha,temp_p,temp_r)

%Calcula o tempo, em segundos, que demora um produto a atingir a
temperatura de refrigeração (0.5ºC acima desta), desde a sua
temperatura inicial. Tem como entradas o tipo e a linha do produto na
respectiva matriz de propriedades e as temperaturas do produto e de
refrigeração, em ºC.

if temp_p==temp_r
    t_r=0;
else

[xm,xp,xfat,xc,xfib,xa,Tempc,cp_r,cp_c,calor_l,kg,maiord,menord,menor2
d,dimcar,lambda,gama1,gama2,N,p1,p2,p3]=propriedades(tipo,linha);
beta1=menor2d/menord;beta2=maiord/menord;dim=dimcar;

%Valores para 0 m/s;
if tipo==1
h=7.7;
end
if tipo==2
h=10.0;
end
if tipo==3
h=23.8;
end
if tipo==4
h=23.8;
end
if tipo==5
h=8;
end
if tipo==6
h=34.3;
end
if tipo==7
h=14.0;
end

k_t=condtermica(tipo,linha,temp_p);
ro=densidade(tipo,linha,temp_r);
temp_c=evalin('base','Temp_c');

if temp_p>Tempc
cp=caloresp_r(tipo,linha)*1e+3;
else
cp=caloresp_c(tipo,linha,temp_c)*1e+3;
end
biot=h*dimcar/k_t;

%O comprimento característico é a menor distância do centro térmico
até à superfície, dimcar.
E0=1.5*(beta1+beta2+beta1^2*(1+beta2)+beta2^2*(1+beta1))/(beta1*beta2*
(1+beta1+beta2))-(((beta1-beta2)^2)^0.4)/15;
Einf=0.75+p1*(1/beta1^2+0.01*p3*exp(beta1-
beta1^2/6))+p2*(1/beta2^2+0.01*p3*exp(beta2-beta2^2/6));
E=(biot^(4/3)+1.85)/(biot^(4/3)/Einf+1.85/E0);
L_inf=1.271+0.305*exp(0.172*gama1-0.115*gama1^2)+0.425*exp(0.09*gama2-
0.128*gama2^2);
j_c=(biot^1.35+1/lambda)/(biot^1.35/L_inf+1/lambda);
niu=((1.5+0.69*biot)/(1.5+biot))^N; j_m=niu*j_c;
```

```
omega_curva=polyfit(evalin('base','valor_omega(:,1)'),evalin('base','v
alor_omega(:,2)'),1);
omega=polyval(omega_curva,biot);

if biot>50
omega=3.1;
end

%A temperatura do produto, t_p, vai ser sempre 0.5ºC maior que a da
envolvente, neste processo.

    Y=-0.5/(temp_r-temp_p);
    t_r=((3*ro*cp*(dim^2))/(omega^2*k_t*E))*log(j_m/Y);

end
end
```

ANEXO B: GUI's

Uma vez que a programação de todos os GUI's das diferentes localizações é idêntica, mudando apenas as chamadas às linhas e colunas das matrizes de dados e resultados, apenas será apresentado aqui o código do GUI dos peixes da 1ª prateleira, os restantes encontram-se publicados na versão pdf deste relatório.

GUI da 1ª prateleira: peixes

```
function varargout = prateleira1_p(varargin)
% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @prateleira1_p_OpeningFcn, ...
                   'gui_OutputFcn',  @prateleira1_p_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT

% --- Executes just before prateleira1_p is made visible.
function prateleira1_p_OpeningFcn(hObject, eventdata, handles,
varargin)
handles.output = hObject;
guidata(hObject, handles);

function varargout = prateleira1_p_OutputFcn(hObject, eventdata,
handles)
varargout{1} = handles.output;


function p1p_hal_Callback(hObject, eventdata, handles)

m_produto=str2double(get(hObject,'String'));
if isnan(m_produto) || m_produto<0
    set(hObject,'String',0);
    errordlg('O valor deve ser um número positivo.','Erro');
end
set(handles.p1p_hal,'UserData',m_produto);
set(handles.p1p_thal,'Enable','on');
evalin('base',sprintf('peixes_dados(3,54)=%u',1));
evalin('base','clc');

function p1p_hal_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function p1p_bac_Callback(hObject, eventdata, handles)

m_produto=str2double(get(hObject,'String'));
if isnan(m_produto) || m_produto<0
    set(hObject,'String',0);
    errordlg('O valor deve ser um número positivo.','Erro');
end
set(handles.p1p_bac,'UserData',m_produto);
set(handles.p1p_tbac,'Enable','on');
evalin('base',sprintf('peixes_dados(1,54)=%u',1));
evalin('base','clc');

function p1p_bac_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function p1p_had_Callback(hObject, eventdata, handles)

m_produto=str2double(get(hObject,'String'));
if isnan(m_produto) || m_produto<0
    set(hObject,'String',0);
    errordlg('O valor deve ser um número positivo.','Erro');
end
set(handles.p1p_had,'UserData',m_produto);
set(handles.p1p_thad,'Enable','on');
evalin('base',sprintf('peixes_dados(2,54)=%u',1));
evalin('base','clc');

function p1p_had_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function p1p_af_Callback(hObject, eventdata, handles)

m_produto=str2double(get(hObject,'String'));
if isnan(m_produto) || m_produto<0
    set(hObject,'String',0);
    errordlg('O valor deve ser um número positivo.','Erro');
end
set(handles.p1p_af,'UserData',m_produto);
set(handles.p1p_taf,'Enable','on');
evalin('base',sprintf('peixes_dados(4,54)=%u',1));
evalin('base','clc');

function p1p_af_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function p1p_cav_Callback(hObject, eventdata, handles)

m_produto=str2double(get(hObject,'String'));
```

```
if isnan(m_produto) || m_produto<0
    set(hObject,'String',0);
    errordlg('O valor deve ser um número positivo.','Erro');
end
set(handles.p1p_cav,'UserData',m_produto);
set(handles.p1p_tcav,'Enable','on');
evalin('base',sprintf('peixes_dados(5,54)=%u',1));
evalin('base','clc');

function p1p_cav_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function p1p_per_Callback(hObject, eventdata, handles)

m_produto=str2double(get(hObject,'String'));
if isnan(m_produto) || m_produto<0
    set(hObject,'String',0);
    errordlg('O valor deve ser um número positivo.','Erro');
end
set(handles.p1p_per,'UserData',m_produto);
set(handles.p1p_tper,'Enable','on');
evalin('base',sprintf('peixes_dados(6,54)=%u',1));
evalin('base','clc');

function p1p_per_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function p1p_tbac_Callback(hObject, eventdata, handles)

valor=get(hObject,'Value');
if valor==1
    errordlg('Escolher a temperatura do produto.','Erro');
    set(handles.p1p_bac,'String',0);
    temp=0; m_produto=0;
end
if valor==2
    temp=evalin('base','Temp');
    m_produto=get(handles.p1p_bac,'UserData');
end
if valor==3
    temp=evalin('base','Temp_r');
    m_produto=get(handles.p1p_bac,'UserData');
end
if valor==4
    temp=evalin('base','Temp_c');
    m_produto=get(handles.p1p_bac,'UserData');
end

if isempty(m_produto)
    errordlg('Introduzir a quantidade e depois a
temperatura.','Erro');
    set(hObject,'Value',1);
```

```
    m_produto=0;
end

evalin('base',sprintf('peixes_dados(1,54)=%u',0));
evalin('base',sprintf('peixes_dados(1,48)=%u',m_produto/(densidade(6,1
,temp))));
if valor==2 || valor==3
    tempo_arref=tempo_r(6,1,temp,evalin('base','Temp_1ap'));
end
if valor==4
    tempo_arref=tempo_c(6,1,evalin('base','Temp_1ap'),temp);
end
evalin('base',sprintf('peixes_dados(1,41)=%u',tempo_arref));
evalin('base',sprintf('peixes_dados(1,13)=%u',m_produto));
evalin('base',sprintf('peixes_dados(1,14)=%u',temp));
evalin('base','clc');

function p1p_tbac_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function p1p_thad_Callback(hObject, eventdata, handles)

valor=get(hObject,'Value');
if valor==1
    errordlg('Escolher a temperatura do produto.','Erro');
    set(handles.p1p_had,'String',0);
    temp=0;
    m_produto=0;
end
if valor==2
    temp=evalin('base','Temp');
    m_produto=get(handles.p1p_had,'UserData');
end
if valor==3
    temp=evalin('base','Temp_r');
    m_produto=get(handles.p1p_had,'UserData');
end
if valor==4
    temp=evalin('base','Temp_c');
    m_produto=get(handles.p1p_had,'UserData');
end
if isempty(m_produto)
    errordlg('Introduzir a quantidade e depois a
temperatura.','Erro');
    set(hObject,'Value',1);
    m_produto=0;
end
evalin('base',sprintf('peixes_dados(2,54)=%u',0));
evalin('base',sprintf('peixes_dados(2,48)=%u',m_produto/(densidade(6,2
,temp))));
if valor==2 || valor==3
    tempo_arref=tempo_r(6,2,temp,evalin('base','Temp_1ap'));
end
if valor==4
    tempo_arref=tempo_c(6,2,evalin('base','Temp_1ap'),temp);
```

```
end
evalin('base',sprintf('peixes_dados(2,41)=%u',tempo_arref));
evalin('base',sprintf('peixes_dados(2,13)=%u',m_produto));
evalin('base',sprintf('peixes_dados(2,14)=%u',temp));
evalin('base','clc');

function p1p_thad_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function p1p_thal_Callback(hObject, eventdata, handles)

valor=get(hObject,'Value');
if valor==1
    errordlg('Escolher a temperatura do produto.','Erro');
    set(handles.p1p_hal,'String',0);
    temp=0;
    m_produto=0;
end
if valor==2
    temp=evalin('base','Temp');
    m_produto=get(handles.p1p_hal,'UserData');
end
if valor==3
    temp=evalin('base','Temp_r');
    m_produto=get(handles.p1p_hal,'UserData');
end
if valor==4
    temp=evalin('base','Temp_c');
    m_produto=get(handles.p1p_hal,'UserData');
end
if isempty(m_produto)
    errordlg('Introduzir a quantidade e depois a
temperatura.','Erro');
    set(hObject,'Value',1);
    m_produto=0;
end
evalin('base',sprintf('peixes_dados(3,54)=%u',0));
evalin('base',sprintf('peixes_dados(3,48)=%u',m_produto/(densidade(6,3
,temp))));
if valor==2 || valor==3
    tempo_arref=tempo_r(6,3,temp,evalin('base','Temp_1ap'));
end
if valor==4
    tempo_arref=tempo_c(6,3,evalin('base','Temp_1ap'),temp);
end
evalin('base',sprintf('peixes_dados(3,41)=%u',tempo_arref));
evalin('base',sprintf('peixes_dados(3,13)=%u',m_produto));
evalin('base',sprintf('peixes_dados(3,14)=%u',temp));
evalin('base','clc');

function p1p_thal_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function p1p_taf_Callback(hObject, eventdata, handles)

valor=get(hObject,'Value');
if valor==1
    errordlg('Escolher a temperatura do produto.','Erro');
    set(handles.p1p_af,'String',0);
    temp=0;
    m_produto=0;
end
if valor==2
    temp=evalin('base','Temp');
    m_produto=get(handles.p1p_af,'UserData');
end
if valor==3
    temp=evalin('base','Temp_r');
    m_produto=get(handles.p1p_af,'UserData');
end
if valor==4
    temp=evalin('base','Temp_c');
    m_produto=get(handles.p1p_af,'UserData');
end
if isempty(m_produto)
    errordlg('Introduzir a quantidade e depois a
temperatura.','Erro');
    set(hObject,'Value',1);
    m_produto=0;
end
evalin('base',sprintf('peixes_dados(4,54)=%u',0));
evalin('base',sprintf('peixes_dados(4,48)=%u',m_produto/(densidade(6,4
,temp))));
if valor==2 || valor==3
    tempo_arref=tempo_r(6,4,temp,evalin('base','Temp_1ap'));
end
if valor==4
    tempo_arref=tempo_c(6,4,evalin('base','Temp_1ap'),temp);
end
evalin('base',sprintf('peixes_dados(4,41)=%u',tempo_arref));
evalin('base',sprintf('peixes_dados(4,13)=%u',m_produto));
evalin('base',sprintf('peixes_dados(4,14)=%u',temp));
evalin('base','clc');

function p1p_taf_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function p1p_tcav_Callback(hObject, eventdata, handles)

valor=get(hObject,'Value');
if valor==1
    errordlg('Escolher a temperatura do produto.','Erro');
    set(handles.p1p_cav,'String',0);
    temp=0;
    m_produto=0;
end
if valor==2
    temp=evalin('base','Temp');
```

```
    m_produto=get(handles.p1p_cav,'UserData');
end
if valor==3
    temp=evalin('base','Temp_r');
    m_produto=get(handles.p1p_cav,'UserData');
end
if valor==4
    temp=evalin('base','Temp_c');
    m_produto=get(handles.p1p_cav,'UserData');
end
if isempty(m_produto)
    errordlg('Introduzir a quantidade e depois a
temperatura.','Erro');
    set(hObject,'Value',1);
    m_produto=0;
end
evalin('base',sprintf('peixes_dados(5,54)=%u',0));
evalin('base',sprintf('peixes_dados(5,48)=%u',m_produto/(densidade(6,5
,temp))));
if valor==2 || valor==3
    tempo_arref=tempo_r(6,5,temp,evalin('base','Temp_1ap'));
end
if valor==4
    tempo_arref=tempo_c(6,5,evalin('base','Temp_1ap'),temp);
end
evalin('base',sprintf('peixes_dados(5,41)=%u',tempo_arref));
evalin('base',sprintf('peixes_dados(5,13)=%u',m_produto));
evalin('base',sprintf('peixes_dados(5,14)=%u',temp));
evalin('base','clc');

function p1p_tcav_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function p1p_tper_Callback(hObject, eventdata, handles)

valor=get(hObject,'Value');
if valor==1
    errordlg('Escolher a temperatura do produto.','Erro');
    set(handles.p1p_per,'String',0);
    temp=0;
    m_produto=0;
end
if valor==2
    temp=evalin('base','Temp');
    m_produto=get(handles.p1p_per,'UserData');
end
if valor==3
    temp=evalin('base','Temp_r');
    m_produto=get(handles.p1p_per,'UserData');
end
if valor==4
    temp=evalin('base','Temp_c');
    m_produto=get(handles.p1p_per,'UserData');
end
if isempty(m_produto)
    errordlg('Introduzir a quantidade e depois a
temperatura.','Erro');
```

```
    set(hObject,'Value',1);
    m_produto=0;
end
evalin('base',sprintf('peixes_dados(6,54)=%u',0));
evalin('base',sprintf('peixes_dados(6,48)=%u',m_produto/(densidade(6,6
,temp))));
if valor==2 || valor==3
    tempo_arref=tempo_r(6,6,temp,evalin('base','Temp_1ap'));
end
if valor==4
    tempo_arref=tempo_c(6,6,evalin('base','Temp_1ap'),temp);
end
evalin('base',sprintf('peixes_dados(6,41)=%u',tempo_arref));
evalin('base',sprintf('peixes_dados(6,13)=%u',m_produto));
evalin('base',sprintf('peixes_dados(6,14)=%u',temp));
evalin('base','clc');

function p1p_tper_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function p1p_pesc_Callback(hObject, eventdata, handles)

m_produto=str2double(get(hObject,'String'));
if isnan(m_produto) || m_produto<0
    set(hObject,'String',0);
    errordlg('O valor deve ser um número positivo.','Erro');
end
set(handles.p1p_pesc,'UserData',m_produto);
set(handles.p1p_tpesc,'Enable','on');
evalin('base',sprintf('peixes_dados(7,54)=%u',1));
evalin('base','clc');

function p1p_pesc_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function p1p_atf_Callback(hObject, eventdata, handles)

m_produto=str2double(get(hObject,'String'));
if isnan(m_produto) || m_produto<0
    set(hObject,'String',0);
    errordlg('O valor deve ser um número positivo.','Erro');
end
set(handles.p1p_atf,'UserData',m_produto);
set(handles.p1p_tatf,'Enable','on');
evalin('base',sprintf('peixes_dados(9,54)=%u',1));
evalin('base','clc');

function p1p_atf_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function p1p_pes_Callback(hObject, eventdata, handles)

m_produto=str2double(get(hObject,'String'));
if isnan(m_produto) || m_produto<0
    set(hObject,'String',0);
    errordlg('O valor deve ser um número positivo.','Erro');
end
set(handles.p1p_pes,'UserData',m_produto);
set(handles.p1p_tpes,'Enable','on');
evalin('base',sprintf('peixes_dados(10,54)=%u',1));
evalin('base','clc');

function p1p_pes_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function p1p_tpesc_Callback(hObject, eventdata, handles)

valor=get(hObject,'Value');
if valor==1
    errordlg('Escolher a temperatura do produto.','Erro');
    set(handles.p1p_pesc,'String',0);
    temp=0;
    m_produto=0;
end
if valor==2
    temp=evalin('base','Temp');
    m_produto=get(handles.p1p_pesc,'UserData');
end
if valor==3
    temp=evalin('base','Temp_1ap');
    m_produto=get(handles.p1p_pesc,'UserData');
end
if valor==4
    temp=evalin('base','Temp_c');
    m_produto=get(handles.p1p_pesc,'UserData');
end
if isempty(m_produto)
    errordlg('Introduzir a quantidade e depois a
temperatura.','Erro');
    set(hObject,'Value',1);
    m_produto=0;
end
evalin('base',sprintf('peixes_dados(7,54)=%u',0));
evalin('base',sprintf('peixes_dados(7,48)=%u',m_produto/(densidade(6,7
,temp))));
if valor==2 || valor==3
    tempo_arref=tempo_r(6,7,temp,evalin('base','Temp_1ap'));
end
if valor==4
    tempo_arref=tempo_c(6,7,evalin('base','Temp_1ap'),temp);
end
evalin('base',sprintf('peixes_dados(7,41)=%u',tempo_arref));
evalin('base',sprintf('peixes_dados(7,13)=%u',m_produto));
evalin('base',sprintf('peixes_dados(7,14)=%u',temp));
evalin('base','clc');

function p1p_tpesc_CreateFcn(hObject, eventdata, handles)
```

```
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function p1p_tatf_Callback(hObject, eventdata, handles)

valor=get(hObject,'Value');
if valor==1
    errordlg('Escolher a temperatura do produto.','Erro');
    set(handles.p1p_atf,'String',0);
    temp=0;
    m_produto=0;
end
if valor==2
    temp=evalin('base','Temp');
    m_produto=get(handles.p1p_atf,'UserData');
end
if valor==3
    temp=evalin('base','Temp_r');
    m_produto=get(handles.p1p_atf,'UserData');
end
if valor==4
    temp=evalin('base','Temp_c');
    m_produto=get(handles.p1p_atf,'UserData');
end
if isempty(m_produto)
    errordlg('Introduzir a quantidade e depois a
temperatura.','Erro');
    set(hObject,'Value',1);
    m_produto=0;
end
evalin('base',sprintf('peixes_dados(9,54)=%u',0));
evalin('base',sprintf('peixes_dados(9,48)=%u',m_produto/(densidade(6,9
,temp))));
if valor==2 || valor==3
    tempo_arref=tempo_r(6,9,temp,evalin('base','Temp_1ap'));
end
if valor==4
    tempo_arref=tempo_c(6,9,evalin('base','Temp_1ap'),temp);
end
evalin('base',sprintf('peixes_dados(9,41)=%u',tempo_arref));
evalin('base',sprintf('peixes_dados(9,13)=%u',m_produto));
evalin('base',sprintf('peixes_dados(9,14)=%u',temp));
evalin('base','clc');

function p1p_tatf_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function p1p_tpes_Callback(hObject, eventdata, handles)

valor=get(hObject,'Value');
if valor==1
    errordlg('Escolher a temperatura do produto.','Erro');
    set(handles.p1p_pes,'String',0);
    temp=0;
```

```
    m_produto=0;
end
if valor==2
    temp=evalin('base','Temp');
    m_produto=get(handles.p1p_pes,'UserData');
end
if valor==3
    temp=evalin('base','Temp_1ap');
    m_produto=get(handles.p1p_pes,'UserData');
end
if valor==4
    temp=evalin('base','Temp_c');
    m_produto=get(handles.p1p_pes,'UserData');
end
if isempty(m_produto)
    errordlg('Introduzir a quantidade e depois a
temperatura.','Erro');
    set(hObject,'Value',1);
    m_produto=0;
end
evalin('base',sprintf('peixes_dados(10,54)=%u',0));
evalin('base',sprintf('peixes_dados(10,48)=%u',m_produto/(densidade(6,
10,temp))));
if valor==2 || valor==3
    tempo_arref=tempo_r(6,10,temp,evalin('base','Temp_1ap'));
end
if valor==4
    tempo_arref=tempo_c(6,10,evalin('base','Temp_1ap'),temp);
end
evalin('base',sprintf('peixes_dados(10,41)=%u',tempo_arref));
evalin('base',sprintf('peixes_dados(10,13)=%u',m_produto));
evalin('base',sprintf('peixes_dados(10,14)=%u',temp));
evalin('base','clc');

function p1p_tpes_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function p1p_am_Callback(hObject, eventdata, handles)

m_produto=str2double(get(hObject,'String'));
if isnan(m_produto) || m_produto<
    set(hObject,'String',0);
    errordlg('O valor deve ser um número positivo.','Erro');
end
set(handles.p1p_am,'UserData',m_produto);
set(handles.p1p_tam,'Enable','on');
evalin('base',sprintf('peixes_dados(11,54)=%u',1));
evalin('base','clc');

function p1p_am_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function p1p_lag_Callback(hObject, eventdata, handles)
```

```
m_produto=str2double(get(hObject,'String'));
if isnan(m_produto) || m_produto<0
    set(hObject,'String',0);
    errordlg('O valor deve ser um número positivo.','Erro');
end
set(handles.p1p_lag,'UserData',m_produto);
set(handles.p1p_tlag,'Enable','on');
evalin('base',sprintf('peixes_dados(12,54)=%u',1));
evalin('base','clc');

function p1p_lag_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function p1p_tam_Callback(hObject, eventdata, handles)

valor=get(hObject,'Value');
if valor==1
    errordlg('Escolher a temperatura do produto.','Erro');
    set(handles.p1p_am,'String',0);
    temp=0;
    m_produto=0;
end
if valor==2
    temp=evalin('base','Temp');
    m_produto=get(handles.p1p_am,'UserData');
end
if valor==3
    temp=evalin('base','Temp_r');
    m_produto=get(handles.p1p_am,'UserData');
end
if valor==4
    temp=evalin('base','Temp_c');
    m_produto=get(handles.p1p_am,'UserData');
end
if isempty(m_produto)
    errordlg('Introduzir a quantidade e depois a
temperatura.','Erro');
    set(hObject,'Value',1);
    m_produto=0;
end
evalin('base',sprintf('peixes_dados(11,54)=%u',0));
evalin('base',sprintf('peixes_dados(11,48)=%u',m_produto/(densidade(6,
11,temp))));
if valor==2 || valor==3
    tempo_arref=tempo_r(6,11,temp,evalin('base','Temp_1ap'));
end
if valor==4
    tempo_arref=tempo_c(6,11,evalin('base','Temp_1ap'),temp);
end
evalin('base',sprintf('peixes_dados(11,41)=%u',tempo_arref));
evalin('base',sprintf('peixes_dados(11,13)=%u',m_produto));
evalin('base',sprintf('peixes_dados(11,14)=%u',temp));
evalin('base','clc');

function p1p_tam_CreateFcn(hObject, eventdata, handles)
```

```
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function p1p_tlag_Callback(hObject, eventdata, handles)

valor=get(hObject,'Value');
if valor==1
    errordlg('Escolher a temperatura do produto.','Erro');
    set(handles.p1p_lag,'String',0);
    temp=0;
    m_produto=0;
end
if valor==2
    temp=evalin('base','Temp');
    m_produto=get(handles.p1p_lag,'UserData');
end
if valor==3
    temp=evalin('base','Temp_r');
    m_produto=get(handles.p1p_lag,'UserData');
end
if valor==4
    temp=evalin('base','Temp_c');
    m_produto=get(handles.p1p_lag,'UserData');
end
if isempty(m_produto)
    errordlg('Introduzir a quantidade e depois a
temperatura.','Erro');
    set(hObject,'Value',1);
    m_produto=0;
end
evalin('base',sprintf('peixes_dados(12,54)=%u',0));
evalin('base',sprintf('peixes_dados(12,48)=%u',m_produto/(densidade(6,
12,temp))));
if valor==2 || valor==3
    tempo_aref=tempo_r(6,12,temp,evalin('base','Temp_1ap'));
end
if valor==4
    tempo_aref=tempo_c(6,12,evalin('base','Temp_1ap'),temp);
end
evalin('base',sprintf('peixes_dados(12,41)=%u',tempo_aref));
evalin('base',sprintf('peixes_dados(12,13)=%u',m_produto));
evalin('base',sprintf('peixes_dados(12,14)=%u',temp));
evalin('base','clc');

function p1p_tlag_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function p1p_ost_Callback(hObject, eventdata, handles)

m_produto=str2double(get(hObject,'String'));
if isnan(m_produto) || m_produto<0
    set(hObject,'String',0);
    errordlg('O valor deve ser um número positivo.','Erro');
end
```

```
set(handles.p1p_ost,'UserData',m_produto);
set(handles.p1p_tost,'Enable','on');
evalin('base',sprintf('peixes_dados(13,54)=%u',1));
evalin('base','clc');

function p1p_ost_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function p1p_vie_Callback(hObject, eventdata, handles)

m_produto=str2double(get(hObject,'String'));
if isnan(m_produto) || m_produto<0
    set(hObject,'String',0);
    errordlg('O valor deve ser um número positivo.','Erro');
end
set(handles.p1p_vie,'UserData',m_produto);
set(handles.p1p_tvie,'Enable','on');
evalin('base',sprintf('peixes_dados(14,54)=%u',1));
evalin('base','clc');

function p1p_vie_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function p1p_cam_Callback(hObject, eventdata, handles)

m_produto=str2double(get(hObject,'String'));
if isnan(m_produto) || m_produto<0
    set(hObject,'String',0);
    errordlg('O valor deve ser um número positivo.','Erro');
end
set(handles.p1p_cam,'UserData',m_produto);
set(handles.p1p_tcam,'Enable','on');
evalin('base',sprintf('peixes_dados(15,54)=%u',1));
evalin('base','clc');

function p1p_cam_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function p1p_tost_Callback(hObject, eventdata, handles)

valor=get(hObject,'Value');
if valor==1
    errordlg('Escolher a temperatura do produto.','Erro');
    set(handles.p1p_ost,'String',0);
    temp=0;
    m_produto=0;
end
if valor==2
```

```
    temp=evalin('base','Temp');
    m_produto=get(handles.p1p_ost,'UserData');
end
if valor==3
    temp=evalin('base','Temp_r');
    m_produto=get(handles.p1p_ost,'UserData');
end
if valor==4
    temp=evalin('base','Temp_c');
    m_produto=get(handles.p1p_ost,'UserData');
end
if isempty(m_produto)
    errordlg('Introduzir a quantidade e depois a
temperatura.','Erro');
    set(hObject,'Value',1);
    m_produto=0;
end
evalin('base',sprintf('peixes_dados(13,54)=%u',0));
evalin('base',sprintf('peixes_dados(13,48)=%u',m_produto/(densidade(6,
13,temp))));
if valor==2 || valor==3
    tempo_arref=tempo_r(6,13,temp,evalin('base','Temp_1ap'));
end
if valor==4
    tempo_arref=tempo_c(6,13,evalin('base','Temp_1ap'),temp);
end
evalin('base',sprintf('peixes_dados(13,41)=%u',tempo_arref));
evalin('base',sprintf('peixes_dados(13,13)=%u',m_produto));
evalin('base',sprintf('peixes_dados(13,14)=%u',temp));
evalin('base','clc');

function p1p_tost_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function p1p_tvie_Callback(hObject, eventdata, handles)

valor=get(hObject,'Value');
if valor==1
    errordlg('Escolher a temperatura do produto.','Erro');
    set(handles.p1p_vie,'String',0);
    temp=0;
    m_produto=0;
end
if valor==2
    temp=evalin('base','Temp');
    m_produto=get(handles.p1p_vie,'UserData');
end
if valor==3
    temp=evalin('base','Temp_r');
    m_produto=get(handles.p1p_vie,'UserData');
end
if valor==4
    temp=evalin('base','Temp_c');
    m_produto=get(handles.p1p_vie,'UserData');
end
if isempty(m_produto)
```

```
    errordlg('Introduzir a quantidade e depois a
temperatura.','Erro');
    set(hObject,'Value',1);
    m_produto=0;
end
evalin('base',sprintf('peixes_dados(14,54)=%u',0));
evalin('base',sprintf('peixes_dados(14,48)=%u',m_produto/(densidade(6,
14,temp))));
if valor==2 || valor==3
    tempo_arref=tempo_r(6,14,temp,evalin('base','Temp_1ap'));
end
if valor==4
    tempo_arref=tempo_c(6,14,evalin('base','Temp_1ap'),temp);
end
evalin('base',sprintf('peixes_dados(14,41)=%u',tempo_arref));
evalin('base',sprintf('peixes_dados(14,13)=%u',m_produto));
evalin('base',sprintf('peixes_dados(14,14)=%u',temp));
evalin('base','clc');

function p1p_tvie_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function p1p_tcam_Callback(hObject, eventdata, handles)

valor=get(hObject,'Value');
if valor==1
    errordlg('Escolher a temperatura do produto.','Erro');
    set(handles.p1p_cam,'String',0);
    temp=0;
    m_produto=0;
end
if valor==2
    temp=evalin('base','Temp');
    m_produto=get(handles.p1p_cam,'UserData');
end
if valor==3
    temp=evalin('base','Temp_r');
    m_produto=get(handles.p1p_cam,'UserData');
end
if valor==4
    temp=evalin('base','Temp_c');
    m_produto=get(handles.p1p_cam,'UserData');
end
if isempty(m_produto)
    errordlg('Introduzir a quantidade e depois a
temperatura.','Erro');
    set(hObject,'Value',1);
    m_produto=0;
end
evalin('base',sprintf('peixes_dados(15,54)=%u',0));
evalin('base',sprintf('peixes_dados(15,48)=%u',m_produto/(densidade(6,
15,temp))));
if valor==2 || valor==3
    tempo_arref=tempo_r(6,15,temp,evalin('base','Temp_1ap'));
end
if valor==4
    tempo_arref=tempo_c(6,15,evalin('base','Temp_1ap'),temp);
```

```
end
evalin('base',sprintf('peixes_dados(15,41)=%u',tempo_arref));
evalin('base',sprintf('peixes_dados(15,13)=%u',m_produto));
evalin('base',sprintf('peixes_dados(15,14)=%u',temp));
evalin('base','clc');

function p1p_tcam_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function p1p_fechar_Callback(hObject, eventdata, handles)

volume_prat=sum(evalin('base','peixes_dados(:,48)'));
volume_local=evalin('base','vol_preenchimento_1ap');
volume_total=volume_prat+volume_local;
evalin('base','clear vol_preenchimento_1ap');
assignin('base','vol_preenchimento_1ap',volume_total);
if any(evalin('base','peixes_dados(:,54)')==1)
    errordlg('Seleccionar todas as temperaturas.','Erro');
else
    if volume_total>0.014
        errordlg('O volume dos produtos excede o da prateleira, rever
as quantidades.','Erro');
    else
        close prateleira1_p;
    end
end


function p1p_apagar_Callback(hObject, eventdata, handles)

evalin('base',sprintf('peixes_dados(:,13)=%u',0));
evalin('base',sprintf('peixes_dados(:,14)=%u',0));
evalin('base',sprintf('peixes_dados(:,41)=%u',0));
evalin('base',sprintf('peixes_dados(:,48)=%u',0));
evalin('base',sprintf('r_1ap_p(:,1)=%u',0));
evalin('base',sprintf('r_1ap_p_temp(:,1)=%u',0));
evalin('base','clc');
close prateleira1_p;


function p1p_sal_Callback(hObject, eventdata, handles)

m_produto=str2double(get(hObject,'String'));
if isnan(m_produto) || m_produto<0
    set(hObject,'String',0);
    errordlg('O valor deve ser um número positivo.','Erro');
end
set(handles.p1p_sal,'UserData',m_produto);
set(handles.p1p_tsal,'Enable','on');
evalin('base',sprintf('peixes_dados(8,54)=%u',1));
evalin('base','clc');

function p1p_sal_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
```

```
end


function p1p_tsal_Callback(hObject, eventdata, handles)

valor=get(hObject,'Value');
if valor==1
    errordlg('Escolher a temperatura do produto.','Erro');
    set(handles.p1p_sal,'String',0);
    temp=0;
    m_produto=0;
end
if valor==2
    temp=evalin('base','Temp');
    m_produto=get(handles.p1p_sal,'UserData');
end
if valor==3
    temp=evalin('base','Temp_r');
    m_produto=get(handles.p1p_sal,'UserData');
end
if valor==4
    temp=evalin('base','Temp_c');
    m_produto=get(handles.p1p_sal,'UserData');
end
if isempty(m_produto)
    errordlg('Introduzir a quantidade e depois a
temperatura.','Erro');
    set(hObject,'Value',1);
    m_produto=0;
end
evalin('base',sprintf('peixes_dados(8,54)=%u',0));
evalin('base',sprintf('peixes_dados(8,54)=%u',0));
evalin('base',sprintf('peixes_dados(8,48)=%u',m_produto/(densidade(6,8
,temp))));
if valor==2 || valor==3
    tempo_arref=tempo_r(6,8,temp,evalin('base','Temp_1ap'));
end
if valor==4
    tempo_arref=tempo_c(6,8,evalin('base','Temp_1ap'),temp);
end
evalin('base',sprintf('peixes_dados(8,41)=%u',tempo_arref));
evalin('base',sprintf('peixes_dados(8,13)=%u',m_produto));
evalin('base',sprintf('peixes_dados(8,14)=%u',temp));
evalin('base','clc');

function p1p_tsal_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```