Utilização do OpenSolver com VBA

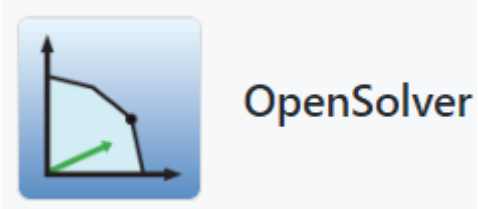
Setembro/2022

Arnaldo Gunzi

Introdução

O OpenSolver é um suplemento poderoso do Excel, para problemas de otimização combinatória.

Foi desenvolvido por Andrew Mason e equipe, da Universidade de Auckland (Nova Zelândia) e pode ser encontrado em: https://opensolver.org.



A ideia básica é que o Solver comum, do Excel, tem um limite de utilização (apenas 200 variáveis). Para problemas maiores do que isso, é necessário adquirir uma licença junto à Frontline, empresa desenvolvedora do Solver.

Já o OpenSolver pode utilizar solver open-source, como o CBC. Ou utilizar ferramentas (pagas) ainda mais poderosas, como o Gurobi ou o CPLEX. Dessa forma, abrimos uma gama de aplicações muito maior do que apenas o Solver da Frontline.

O OpenSolver tem um menu de utilização bastante similar ao Solver comum.

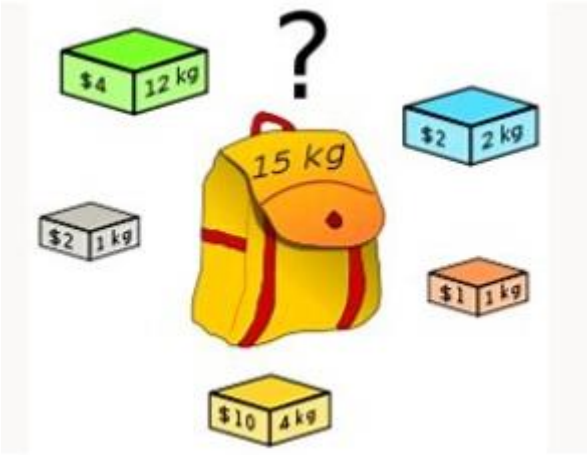
Este tutorial mostra como utilizar VBA para acessar e manipular o OpenSolver.

Qual a vantagem de utilizar o VBA? É ter uma flexibilidade maior na formulação, e com isso poder transformar a planilha numa ferramenta – para o usuário leigo, é só clicar num botão e resolver.

O Problema da Mochila

O Problema da Mochila é um clássico da otimização combinatória. Vamos utilizar como exemplo.

Vou fazer uma viagem, e tenho uma mochila. Há diversos itens que posso escolher para levar. Cada item tem um peso e um valor.



Quais itens devo levar?

Quero maximizar o valor que estou levando na mochila, restrito ao peso máximo que consigo carregar.

No Excel, o peso máximo da mochila e os valores e pesos por item devem ser dados de entrada do problema.

O campo “Solução” deve conter 0 ou 1 (indicando não levar ou levar o item). O Peso Total é a soma do peso dos itens escolhidos, e a Função Objetivo é a soma dos valores dos itens escolhidos.

W	3.470

Peso Total	3.470
Função Objetivo	110.109

i	Valor	Peso
1	978	20
2	552	41
3	471	85
4	438	29
5	520	68
6	846	34
7	612	7
8	229	86
9	947	2
10	594	77
11	717	100
12	196	64
13	221	59
14	403	5
15	684	20
16	918	33

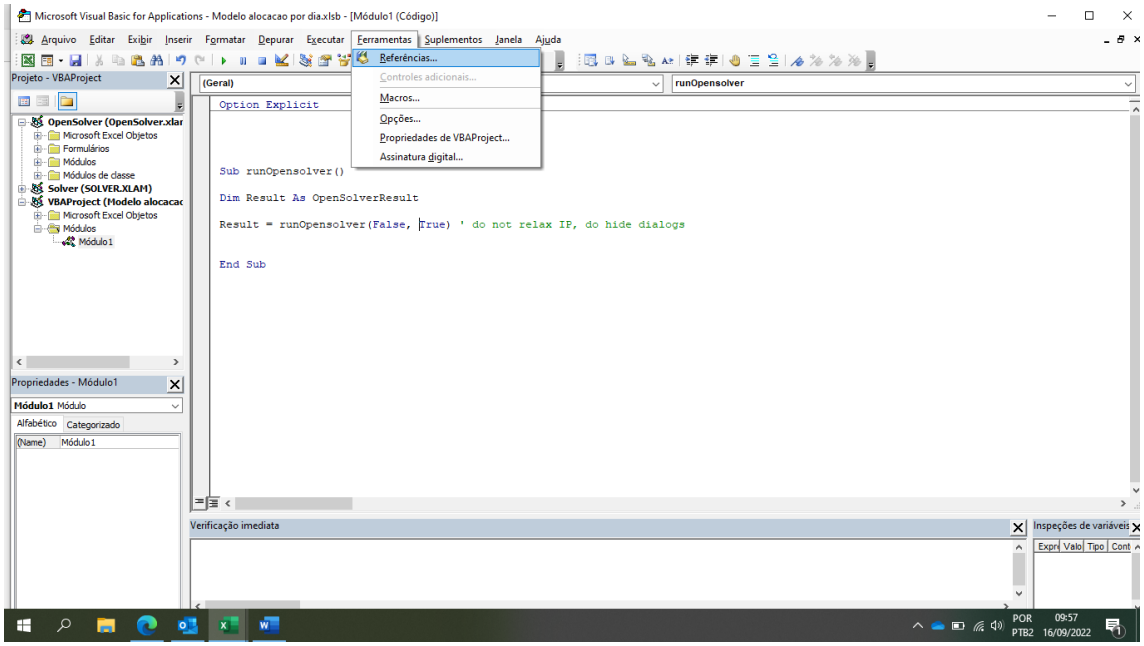
i	Solução
1	1
2	0
3	0
4	0
5	0
6	1
7	1
8	0
9	1
10	0
11	0
12	0
13	0
14	1
15	1
16	1

Baixe o arquivo Excel em Knapsack_OpenSolver.xlsm

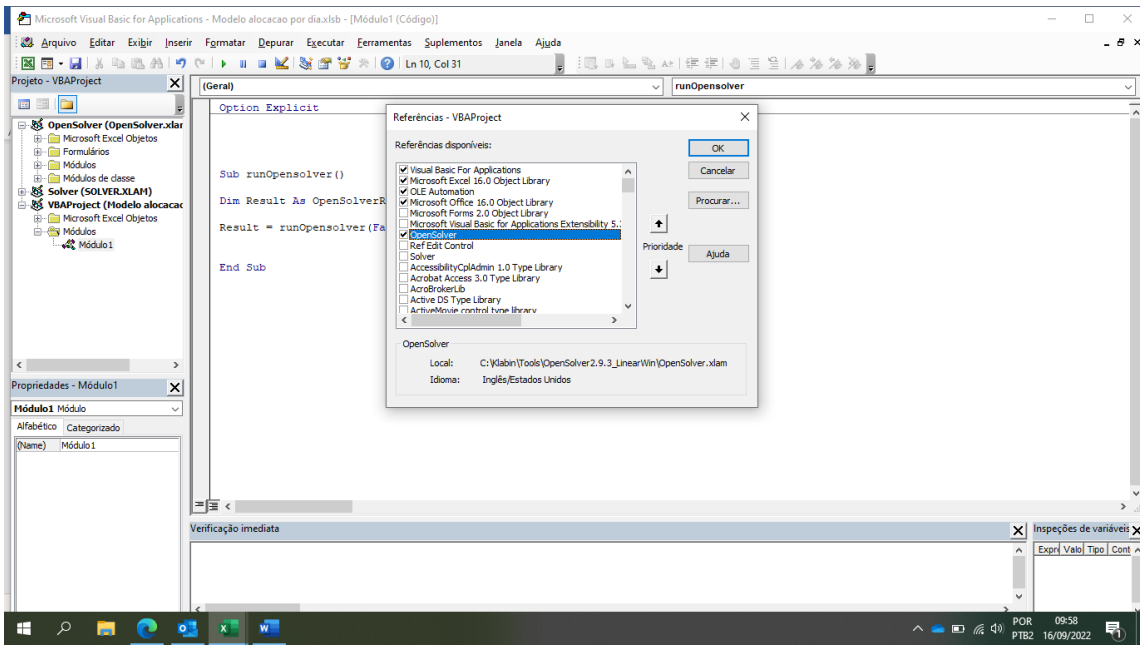
Primeiro passo: adicionar o OpenSolver nas referências do VBA

Como adicionar o OpenSolver nas referências do VBA?

Ir em Ferramentas -> Referências



Adicionar o OpenSolver e clicar em OK.



Como modelar um problema no OpenSolver com VBA

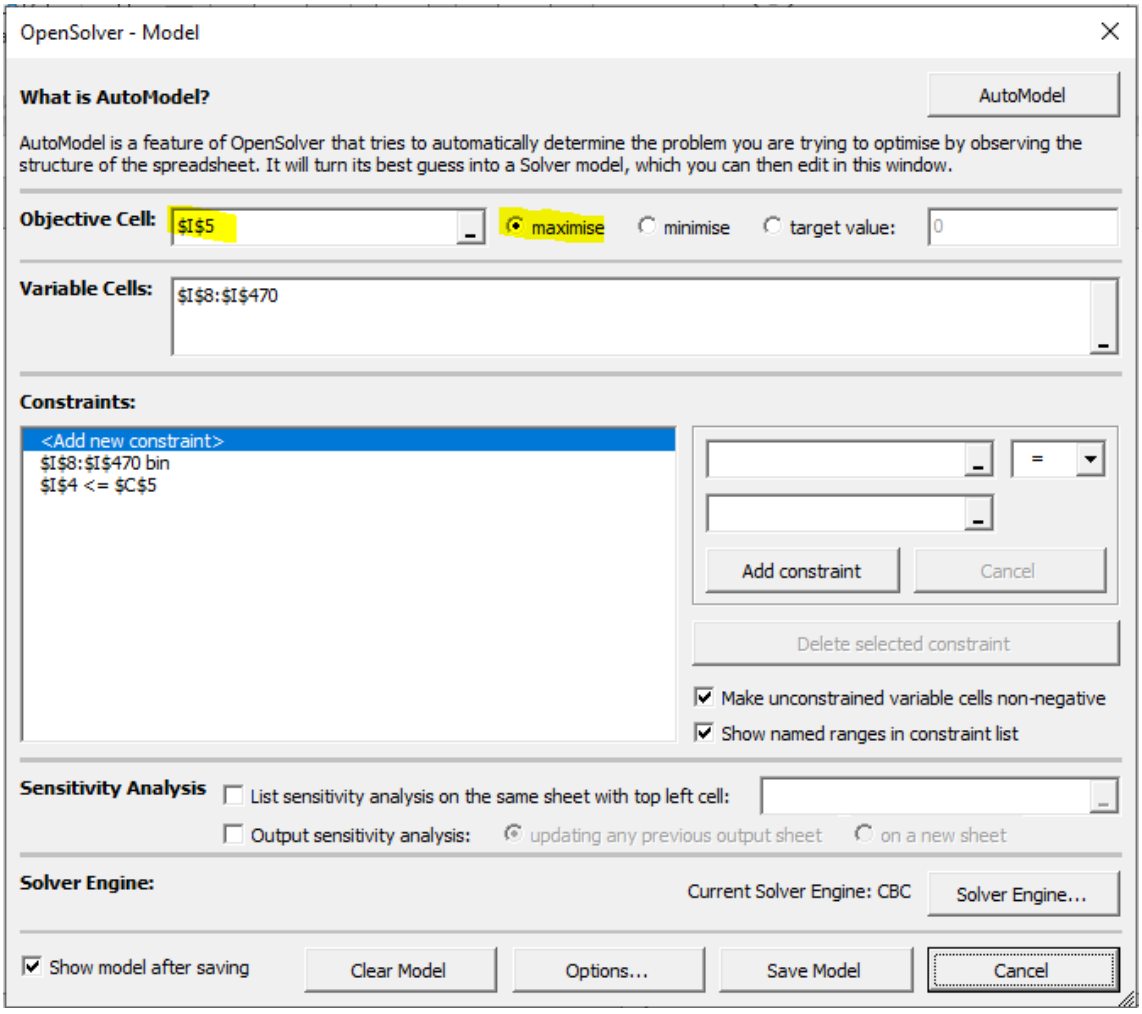
Vamos utilizar o OpenSolver via VBA, para resolver o problema da mochila.

É importante notar que todo comando que fazemos manualmente tem o equivalente via VBA.

A referência para os comandos pode ser consultada em https://opensolver.org/opensolver-api-reference. Outra forma é consultando direto o código VBA do OpenSolver, ele detalha bem a forma de utilização.

Função Objetivo

No painel do OpenSolver, tem um campo para definir a célula com a função objetivo, e a direção: maximizar, minimizar, ou valor alvo.



O equivalente no VBA é a função SetObjectiveFunctionCell:

SetObjectiveFunctionCell Range("i5")

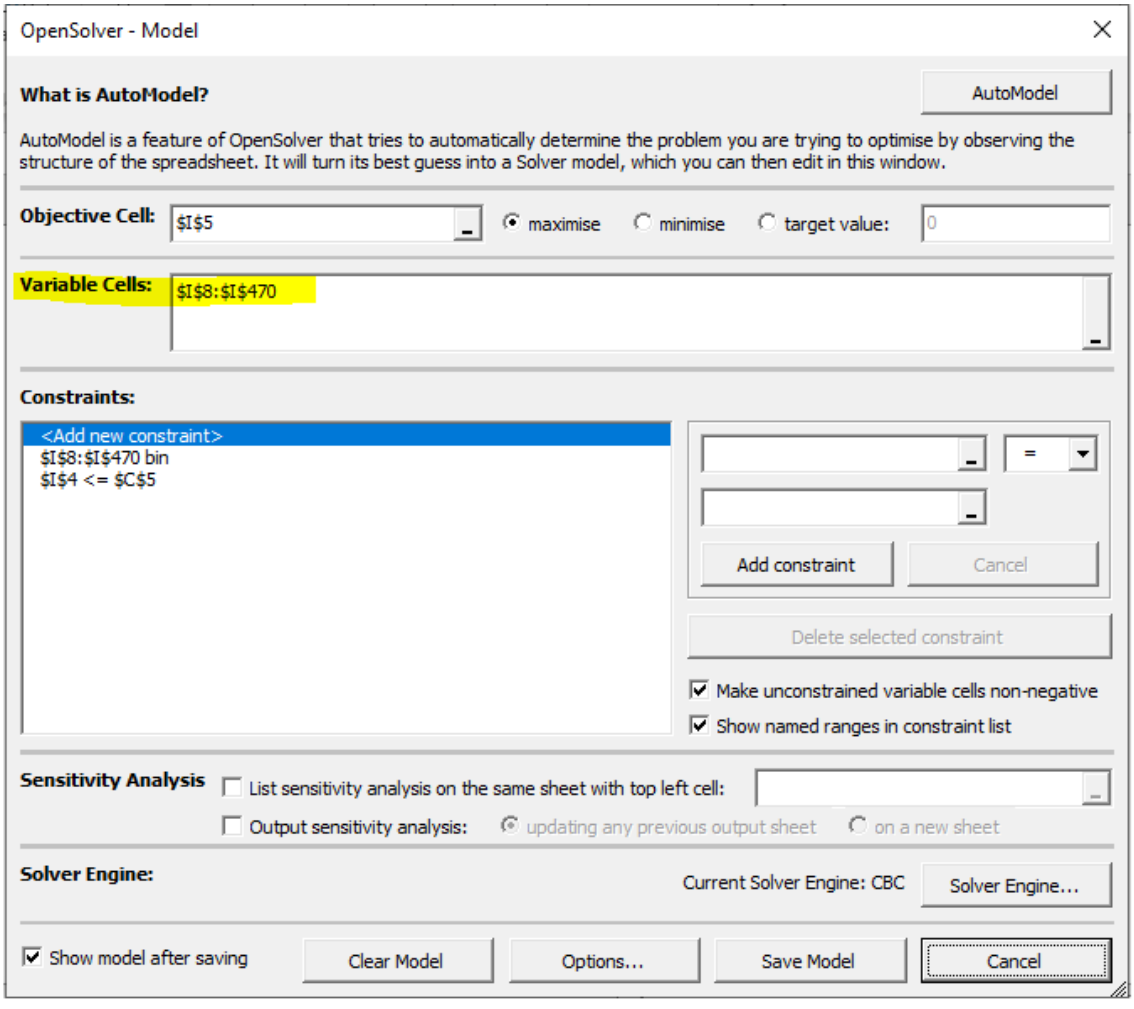
Para minimizar ou maximizar, SetObjectiveSense:

SetObjectiveSense MaximiseObjective

O próprio código, via autocompletar, vai dar as dicas das opções de preenchimento.

Definição de Variáveis

A seguir, podemos definir as variáveis, utilizando o painel do OpenSolver.



O equivalente, via código, é:

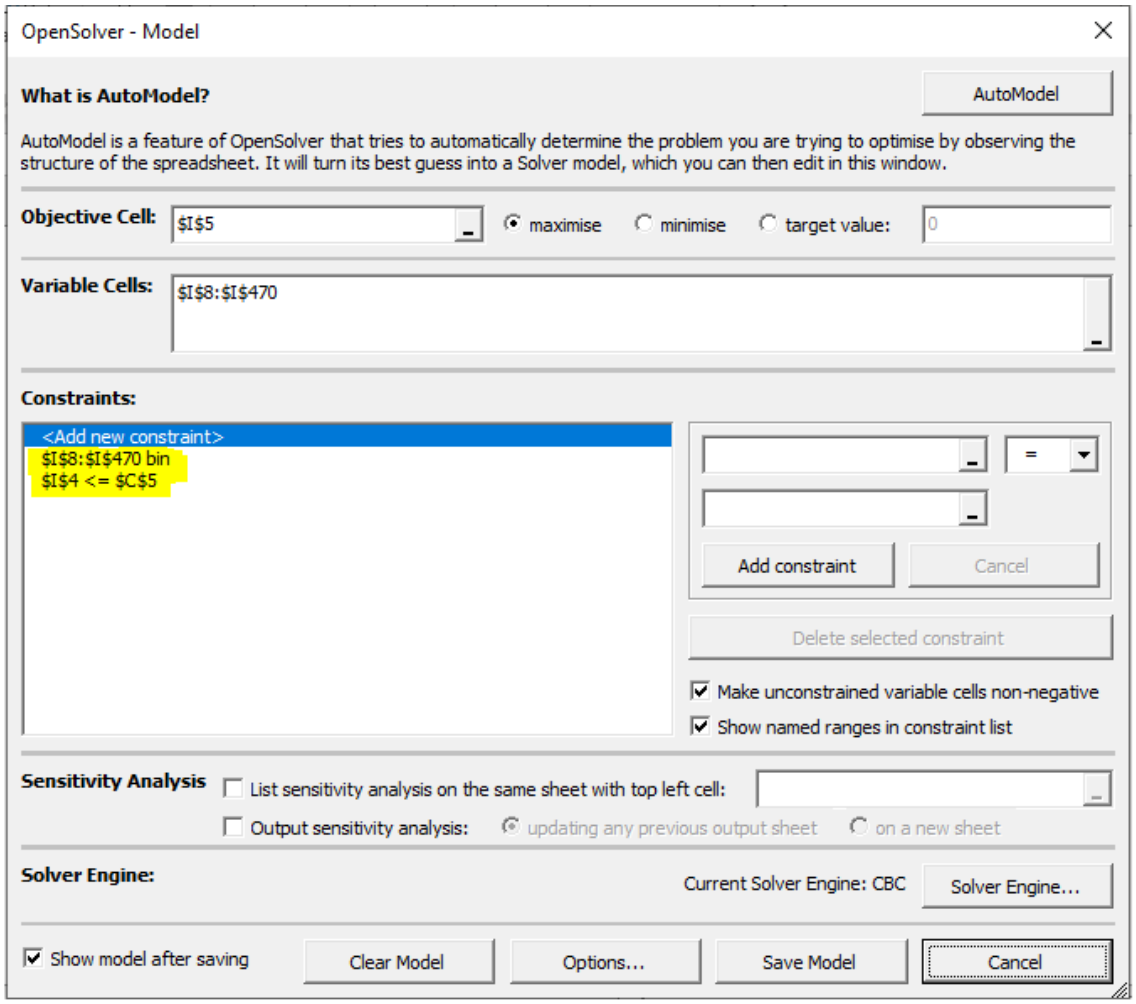
SetDecisionVariables Range("i8:i470")

E é nesse ponto que o código começa a ficar poderoso. Ao invés do range ser fixo e ser mudado a cada novo cenário, como no exemplo, podemos estabelecer o range de acordo com o tamanho do problema modelado via código.

Definição de Restrições

Para o caso em questão, o primeiro ponto a notar é que as variáveis de decisão devem ser binárias.

E a segunda restrição é a de que o valor carregado deve ser menor do que a capacidade máxima da mochila.



O equivalente VBA é bastante similar, com a utilização do comando AddConstraint:

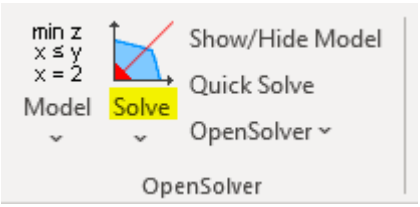
AddConstraint Range("i8:i470"), RelationBIN

AddConstraint Range("i4"), RelationLE, Range("c5")

A notar o parâmetro “RelationLE”, que significa “menor igual”. Há parâmetros para “maior igual”, “igual”, etc, basta consultar as opções no próprio código ou no link de referência dado acima.

Rodar o modelo

Com o modelo formulado, o próximo passo é rodar. Manualmente, isso equivale a clicar no ícone correspondente.



Em código, é utilizar o comando RunOpenSolver:

Dim result As OpenSolverresult

result = RunOpenSolver

O valor retornado será 0, no caso de rodar corretamente, e códigos outros conforme instruções abaixo.

Enum OpenSolverResult
Pending = -4 ‘ Used for solvers that asynchronously and are yet to run
AbortedThruUserAction = -3 ‘ Used to indicate that a non-linearity check was made (losing the

solution)
ErrorOccurred = -2 ‘ Indicate an error occured and has been reported to the user
Unsolved = -1 ‘ Indicates a model not yet solved
Optimal = 0
Unbounded = 4 ‘ objective can be made infinitely good
Infeasible = 5 ‘ There is no solution that satisifies all the constraints
LimitedSubOptimal = 10 ‘ CBC stopped before finding an optimal/feasible/integer solution because of CBC errors or time/iteration limits
NotLinear = 7 ‘ Report non-linearity so that it can be picked up in silent mode
End Enum

Uma última dica é resetar o modelo antes do início da macro:

ResetModel

Senão for feito isso, as restrições antigas vão continuar existindo sobrepondo com as novas, o que pode fazer o modelo ficar incorreto.

Conclusão

O OpenSolver é uma ferramenta poderosa. Aliada ao VBA, pode permitir um grau de automação enorme, e resolver uma gama de problemas de Pesquisa Operacional nas empresas e na vida real.

Há diversas outras opções mais avançadas, a fim de permitir modelos complexos. A ideia do tutorial foi fazer o exemplo mais simples e didático possível.

O OpenSolver, em geral, é recomendado para problemas de tamanho médio. Para problemas muito grandes, começa a ter lentidão demasiada na leitura de dados e formulação do problema, pela forma com que este foi construído (fortemente baseada em Excel).

Exemplo prático: um trabalho de otimização de silvicultura, com cerca de 100 mil variáveis, demorava 6 horas para resolver com OpenSolver puro. Migrando para Python (Pyomo) + CBC, o tempo caiu para 40 min, para resposta de mesma qualidade. Com Python (Pyomo) + Gurobi, o tempo caiu para 4 minutos, para a mesma resposta.

De qualquer forma, cabe ao projetista escolher a melhor ferramenta para o processo em questão, dadas as restrições (custo, tempo de processamento, maturidade do processo). E o

OpenSolver figura como um excelente candidato a resolver problemas difíceis, de forma rápida e flexível.

Referências:

https://opensolver.org/

https://opensolver.org/opensolver-api-reference/

https://en.wikipedia.org/wiki/Knapsack_problem