Application created in COMSOL Multiphysics 6.4

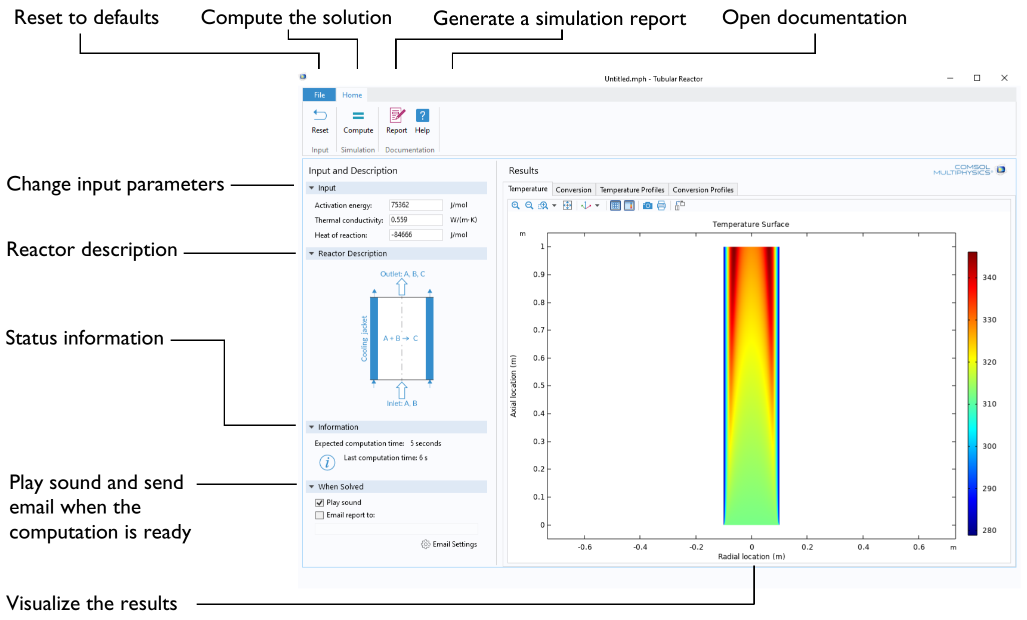
Tubular Reactor

About the Tubular Reactor Application

With this application, students in chemical engineering can model a nonideal tubular reactor, including radial and axial variations in temperature and composition, as well as investigate the impact of different operating conditions. The process described is the exothermic reaction of propylene oxide with water to form propylene glycol.

The application also exemplifies how teachers can build tailored interfaces for problems that challenge the students' imaginations. The model and exercise are originally described in Scott Fogler's *Elements of Chemical Reaction Engineering* (Ref. 1).

The mathematical model consists of an energy balance and a material balance described in an axisymmetric coordinate system. The students can change the activation energy of the reaction, the thermal conductivity, and the heat of reaction in the reactor, see Figure 1. The resulting solution gives the axial and radial conversion as well as temperature profiles in the reactor. For some data, the results from the simulation are not obvious, which means that the interpretation of the model results also becomes a problem-solving exercise.



Activation energy: 75362 J/mol
Thermal conductivity: 0.559 W/(m·K)
Heat of reaction: -84666 J/mol

Expected computation time: 5 seconds
Last computation time: 6 s

☑ Play sound
☐ Email report to:

Figure 1: The application's user interface including the description of the main steps.

The main steps in the use of the application are the following:

- Enter the input parameters, which are the activation energy, the thermal conductivity of the reactor solution, and the heat of reaction. The students may also reset the input parameters to their default values by selecting **Reset**.
- Click the **Compute** button to run the simulation with the given input.
- Click the different **Results** tabs to visualize the temperature and conversion field plots. Also cross-section plots of the temperature and conversion profiles are available in their respective tabs.
- Click the **Report** button to generate a report for a simulation including the input data and the corresponding results.
- Click the **Help** button to read the documentation about the application.

Note that you can have the app play a sound when the simulation has finished by selecting the **Play sound** checkbox under **When Solved**. In that section you can also send an email when the computation is ready by selecting the **Email report to** checkbox and entering an email address in the corresponding text field. If needed, click the **Email Settings** button to open the **Outgoing Server (SMTP)** dialog box. With the preference settings, the app will use either the email preferences set in COMSOL Multiphysics when using the **Test Application** feature or the preferences in COMSOL Server if the app is run on COMSOL Server. Select the **Override preferences** checkbox to specify a host and port, connection security, a username and password, and a from email address. Those settings are saved in the app. This sends a report with the settings and the computed results. The functionality can be used by students to send the results to a supervisor. For computations that take a longer time to compute, this functionality may be of great use. For example, you can start a simulation and leave the office, or laboratory, and then get the full report from the app when the computation is done, which you can access on the road or wherever you have access to your mail.

This app also demonstrates the possibility to localize apps. English, German, and Simplified Chinese localizations are included. The app uses the **From preference** option in the **Language** list in the **Main Window** node's **Settings** window. It then uses the language set for the COMSOL Desktop in the **Preferences** window; if it is another language than the supported ones, English is used as the fallback language.

The Embedded Model

The process described by the embedded model is an exothermic reaction of propylene oxide and water that forms propylene glycol. This reaction takes place in a tubular reactor equipped with a cooling jacket in order to avoid explosion.

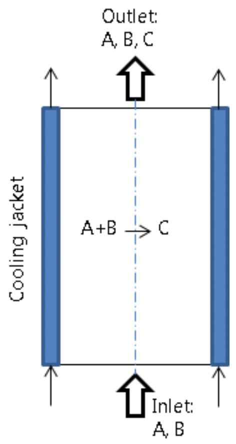


Figure 2: Sketch of the reactor geometry with the cooling jacket.

The reaction takes place in the liquid phase and in the presence of a solvent. The density of the reactor solution is therefore assumed to vary to a negligible extent, despite variations in composition and temperature. Under these assumptions, it is possible to define a fully developed velocity profile along the radius of the reactor.

MODEL DEFINITION

The reaction is a reversible conversion of species A, B, and C in liquid.

$$A + B \Leftrightarrow C$$

A is the notation for propylene oxide, B water, and C propylene glycol. The reaction kinetics is 1st order in regard to the concentration of A, assuming that water is available to such a large excess that its concentration is constant. Further, assuming identical transport properties for the propylene oxide and propylene glycol, and that these species are present in a dilute water solution, makes it possible to approximate the concentration of propylene glycol, species C from the concentration of propylene oxide, species A, and the stoichiometry of the reaction.

The model equations describe the conservation of material and energy. The dependent variables are the concentration, c, and the temperature, T, in the reactor. A third dependent variable for the temperature in the cooling jacket, T_j, is defined along the length of the reactor at the position of the cooling jacket.

The material and energy equations in the reactor are defined along two independent variables: the variable for the radial direction, r, and along the axial direction, z. These equations form a system of two coupled partial differential equations (PDE), along r and z.

The equation for the conservation of material in the reactor is thus the following:

$$D_{\text{eff}}\frac{1}{r}\frac{\partial c_A}{\partial r} + D_{\text{eff}}\frac{\partial^2 c_A}{\partial r^2} + D_{\text{eff}}\frac{\partial^2 c_A}{\partial z^2} - 2U\left(1 - \left(\frac{r}{R}\right)^2\right)\frac{\partial c_A}{\partial z} + r_A = 0$$

where D_{eff} denotes the effective diffusion coefficient, c_A the concentration of species A, U the average flow velocity, R the radius of the reactor, and r_A the reaction rate.

The equation for the conservation of energy in the reactor is the following:

$$k\frac{1}{r}\frac{\partial T}{\partial r} + k\frac{\partial^2 T}{\partial r^2} + k\frac{\partial^2 T}{\partial z^2} - 2U\left(1 - \left(\frac{r}{R}\right)^2\right)\rho C_P\frac{\partial T}{\partial z} - r_A(-\Delta H_{Rx}) = 0$$

where k denotes the thermal conductivity, T temperature, ρ density, C_P the heat capacity, and ΔH_{Rx} the heat of reaction.

The temperature of the cooling jacket is assumed to vary only in the axial direction, that is along the length of the reactor. The energy equation for the cooling jacket is therefore formulated along the z-coordinate:

$$\frac{\partial T_j}{\partial z} = \frac{2\pi R U_k(T - T_j)}{m_J C_{PJ}}$$

where m_J denotes the mass flow rate of the coolant, C_{PJ} its heat capacity, and U_k the heat transfer coefficient between the reactor and the cooling jacket.

The boundary conditions define the concentration and temperature at the inlet of the reactor. At the outlet, the outwards flux of material and energy is dominated by advection and is described accordingly. At the reactor wall, the heat flux is proportional to the temperature difference between the reactor and the cooling jacket:

$$-\frac{\partial T}{\partial r}(R, z) = \frac{U_k}{k}(T - T_j)$$

RESULTS

The results from the simulations are quite interesting. For example, for the default input, the conversion profiles along the radial cut lines display a minimum and a maximum, as shown in Figure 3 below. In Fogler's book, one of the tasks for the student is to explain these profiles.

Here, we can reveal that the profile is explained by the combination of the exothermic reaction, the advective term, and the cooling from the jacket.

In the middle of the reactor, the large flow velocity reduces the conversion, since the reactants reach far into the reactor before they react (labeled 1 in Figure 3).

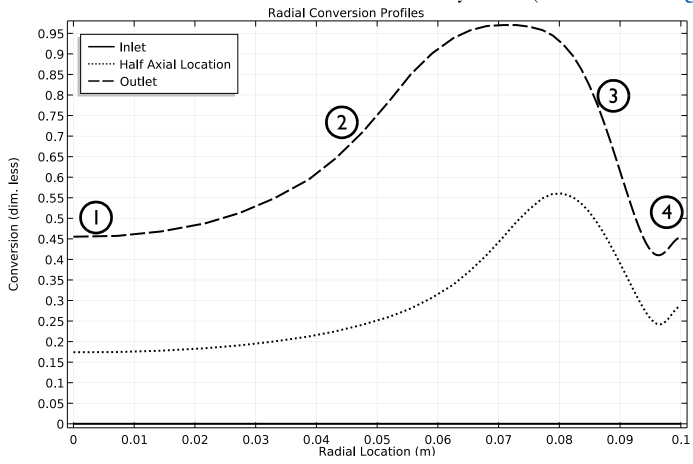


Figure 3: Cross-section plot of the conversion of A along the radius of the reactor and at different positions along the length. The conversion is 1 when all propylene oxide has been converted to propylene glycol at a give point.

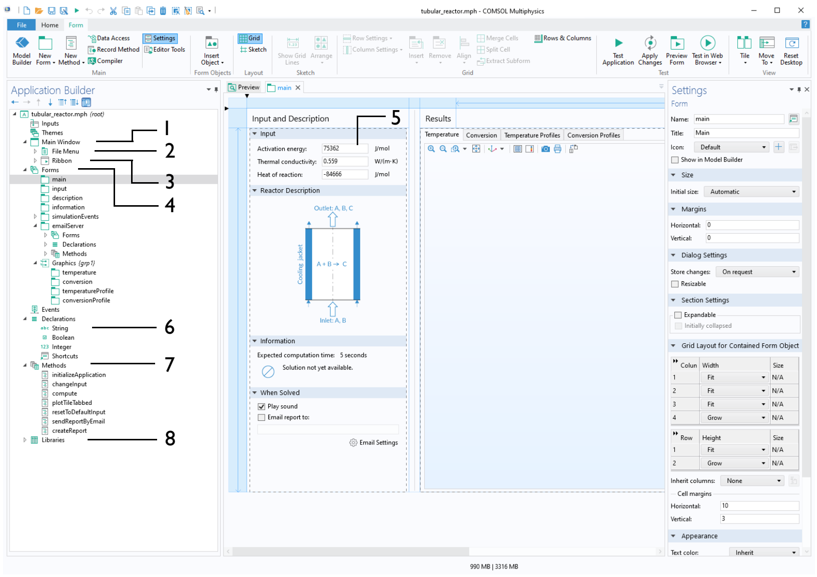
Closer to the wall, the flow rate decreases and the conversion then increases, since the temperature is still relatively high far from the jacket wall, which also gives a high reaction rate (2).

However, as we get even closer to the wall, the conversion starts to decrease due to the cooling of the jacket, which decreases the reaction rate (3) in Figure 3 above.

At the reactor wall, the cooling is very efficient, which should decrease the conversion even more. However, the conversion increases slightly, since there is no advection of reactants at the wall. In other words, the space time for the volume elements that travel at the wall is very high, since the flow is zero at the wall (4). The reactants are therefore consumed to a larger extent.

Notes About the Implementation

For the teacher, the Application Builder tree and the member form preview in the Application Builder reveal the structure of the application; see Figure 4 below. The **Main** node (labeled 1 in Figure 4) contains the child nodes that describe the **File** menu (2) and the **Ribbon** (3). In Linux® and macOS operating systems, the ribbon is shown as a toolbar. It also contains a reference to the main form.



tubular_reactor.mph *(root)*
- Inputs
- Themes
- Main Window
  - File Menu
  - Ribbon
- Forms
  - main
  - input
  - description
  - information
  - simulationEvents
  - emailServer
    - Forms
    - Declarations
    - Methods
  - Graphics *{grp1}*
    - temperature
    - conversion
    - temperatureProfile
    - conversionProfile
- Events
- Declarations
  - String
  - Boolean
  - Integer
  - Shortcuts
- Methods
  - initializeApplication
  - changeInput
  - compute
  - plotTileTabbed
  - resetToDefaultInput
  - sendReportByEmail
  - createReport
- Libraries

Input and Description

Input

Activation energy:	75362	J/mol
Thermal conductivity:	0.559	W/(m·K)
Heat of reaction:	-84666	J/mol

Reactor Description

Information

Expected computation time: 5 seconds

Solution not yet available.

When Solved

☑ Play sound
☐ Email report to:

Email Settings

Settings
Form

Name: main
Title: Main
Icon: Default
☐ Show in Model Builder

Size
Initial size: Automatic

Margins
Horizontal: 0
Vertical: 0

Dialog Settings
Store changes: On request
☐ Resizable

Section Settings
☐ Expandable
☐ Initially collapsed

Grid Layout for Contained Form Object

Colun	Width	Size
1	Fit	N/A
2	Fit	N/A
3	Fit	N/A
4	Grow	N/A

Row	Height	Size
1	Fit	N/A
2	Grow	N/A

Inherit columns: None
Cell margins
Horizontal: 10
Vertical: 3

Appearance
Text color: Inherit

Figure 4: The user interface of the Application Builder containing the Tubular Reactor application settings.

The **Forms** node (4) contains ten forms in this case: One form that describes the main form, one that describes the **Input** section, one that includes the figure in the **Reactor Description** section, one form for the **Information** section displaying the status of the solution and the expected simulation time, one form for handling the events after the simulation, one form for handing the email, and finally four forms that describe the different members in the graphics form collection. These four graphics form members correspond to the temperature field plot, the conversion field plot, the temperature cross-section plot and the conversion cross-section plot, respectively.

The **Input Field** form objects for the activation energy, the thermal conductivity, and the heat of reaction (5) are linked to the corresponding parameters in the embedded model. The range of values is also limited in order to provide a safe input range that does not produce garbage.

The **Declarations** node (6) includes the declaration of variables that are not defined in the embedded model. For instance, you can declare a string variable that displays a message in the user interface when the app is run based on a selection by the user. For example, a string variable is created to show if the simulation results are updated or not (that is, if the student changes the activation energy without re-solving the model equation, a message is shows that warns the student that the shown solution does not corresponds with the settings).

The application further contains a set of **Methods** (7) that correspond to initializing the app, handling changing inputs, computing the results in a simulation, creating the tabbed plots, reset to default values, generating and sending the report. These methods are linked to the corresponding menus in the ribbon or in the main menu.

The **Library** node (8) contains files that are embedded in the application. In this example, we have a PDF-file that contains the application's documentation (this text) linked to the corresponding ribbon menus.

The Tubular Reactor example shows how to create a dedicated user interface based on a model — an application — where students can build an intuitive connection between a physical description of a reactor and the implications of this description in the operation of the reactor. An important component in this exercise is that the results are not obvious; the interpretation of the results requires some thinking.

The Application Builder provides a user-friendly tool for the teacher to graphically create application interfaces. It allows teachers to concentrate on the exercise itself rather than investing time in explaining software tools or programming interfaces in the traditional way. They can focus on generating simulation results that trigger thinking.

The students get more challenging and entertaining exercises that focus on the problem, not on the technicalities of running simulation software.

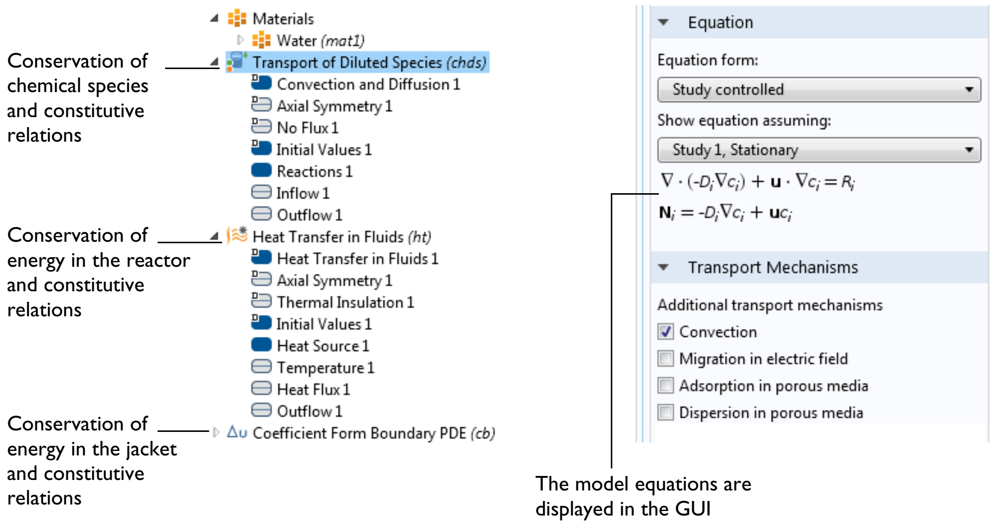
Reference

1. S. Fogler, "Example W12-8 Tubular reactor with axial and radial gradients," *Elements of Chemical Reaction Engineering*, 5th ed., Prentice Hall, p. 601, 2016.

Application Library path: `COMSOL_Multiphysics/Applications/tubular_reactor`

Building the Embedded Model

The Model Definition paragraph above describes the mathematical model used in the application's embedded model. The material and energy balances can be expressed as a system of partial differential equations (PDEs). These equations can be derived by combining the equations for the conservation of mass and energy with the constitutive relations for the flux of mass and energy. The PDEs and their boundary conditions define the mathematical model; that is, they are the mathematical model equations. Solving these equations would give us the concentration and temperature fields in the reactor.


Conservation of chemical species and constitutive relations
Conservation of energy in the reactor and constitutive relations
Conservation of energy in the jacket and constitutive relations
The model equations are displayed in the GUI

- Materials
  - Water *(mat1)*
- Transport of Diluted Species *(chds)*
  - Convection and Diffusion 1
  - Axial Symmetry 1
  - No Flux 1
  - Initial Values 1
  - Reactions 1
  - Inflow 1
  - Outflow 1
- Heat Transfer in Fluids *(ht)*
  - Heat Transfer in Fluids 1
  - Axial Symmetry 1
  - Thermal Insulation 1
  - Initial Values 1
  - Heat Source 1
  - Temperature 1
  - Heat Flux 1
  - Outflow 1
- Coefficient Form Boundary PDE *(cb)*

Equation

Equation form: Study controlled

Show equation assuming: Study 1, Stationary

$$\nabla\cdot(-D_i\nabla c_i) + \mathbf{u}\cdot\nabla c_i = R_i$$

$$\mathbf{N}_i = -D_i\nabla c_i + \mathbf{u}c_i$$

Transport Mechanisms

Additional transport mechanisms

- ✓ Convection
- ☐ Migration in electric field
- ☐ Adsorption in porous media
- ☐ Dispersion in porous media

Figure 5: The model equations are formulated by ready-made physics interfaces. We only need to enter transport and reaction properties.

THE NUMERICAL MODEL

However, the mathematical model equations cannot be solved in an exact analytical way in this case. Instead, we approximate the system of equations with a discretized system of equations using a finite element formulation. As a results, we obtain a numerical model with its corresponding numerical model equations. Note though that the numerical model is an approximation of the mathematical model.

The numerical model equations are obtained by chopping up the model domain — that is, the geometry representing the reactor — in smaller elements. In these elements, the integral form of the PDEs are discretized to give a set of algebraic equations for every element. The algebraic equations for each element are coupled to the algebraic equations for neighboring elements. The elements at the boundaries also get contributions from the boundary conditions. In this way, a system of algebraic equations is formulated which form the numerical model equations.

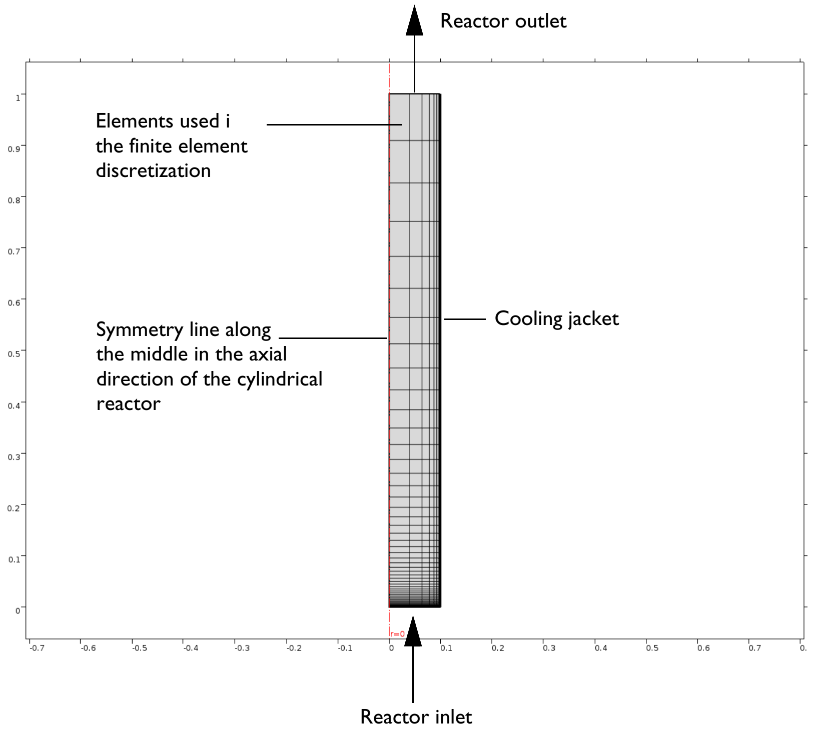

Reactor outlet
Elements used i the finite element discretization
Symmetry line along the middle in the axial direction of the cylindrical reactor
Cooling jacket
Reactor inlet
r=0

Figure 6: The elements used in the formulations of the numerical model are smaller close to the inlet of the reactor and close to the cooling jacket. The reason is that a finer mesh is needed in order to resolve the steepest gradient, which are found at the inlet for the concentration and at the position of the cooling jacket for the temperature field.

If the mathematical model is stable and has a unique solution, and in addition, the numerical model is consistent and stable, then the solution to the numerical model equations should approach the solution of the mathematical model equations as we decrease the size of the elements.

In COMSOL Multiphysics, we can enter transport and reaction properties for the mass and energy equations directly in the user interface (UI); see Figure 6. The package automatically formulates a system of PDEs based on our input, discretizes this system to generate the numerical model equations, and solves these numerical model equations when we click the **Compute** button.

Model Verification

As we mentioned above, the numerical model equations are approximations of the mathematical model equations. In general, it may not be easy to verify that the solution of the numerical model equations give an accurate representation of the solution to the mathematical model equations. However, there are different methods that may give us estimates of the accuracy of the numerical solution in representing the mathematical solution to the model equations. Model verification is an important step in estimating the accuracy of this representation.

The first step is to investigate the influence of the solver settings on the solution. If the tolerances are too coarse then they can limit the accuracy of the numerical model. The point of doing solver tolerance variation is to exclude this possibility or reduce this risk. Another potential problem is when the numerical model or method is not stable, which can be revealed by varying the tolerances.

SOLVER TOLERANCES IN THE NUMERICAL SOLUTION

To estimate the influence of the solver settings and tolerances, we may choose to study quantities that are of central interest and which can be difficult to compute.

If we look at our reactor model, the conservation of chemical species is one of the basic concepts included in the numerical model. The solution should therefore capture this important principle. Although the flux in the reactor is easy to compute, the reaction term includes exponential dependence of temperature and should therefore be sensitive to tolerances.

We can use the knowledge about the conservation of mass to check the solver tolerances. At steady state, the amount of A reacting in the reactor should be balanced by the difference in the flow of this species in and out of the reactor. If we see a large error, the tolerance is set to high or there is something wrong in the implementation of the numerical equations.

The total reaction rate in the reactor is expressed by the following integral over the reactor domain:

$$R_A = \int_{\Omega} (r_{A,h} dx)$$

where $r_{A,h}$ denotes the numerically computed reaction rate in mole/(m^3·s) and depends exponentially on the temperature thorough the Arrhenius equation. We can compute this directly in the COMSOL Multiphysics user interface by, for example, using derived values for surface integration (surface integrations since our model is axisymmetric).

The total feed rate of species A at the inlet is given by the following boundary integral:

$$F_{A,0} = \int_{\partial\Omega_{in}} c_{A,0} u_z dS$$

where u_z denotes the velocity component in the axial (z) direction in the reactor, which is perpendicular to both the inlet and outlet boundaries. The outflow is thus given by:

$$F_A = \int_{\partial\Omega_{out}} c_{A,h} u_z dS$$

Note that $r_{A,h}$ denotes the numerically computed concentration field. These two integrals can be computed using derived values and line integrals in the COMSOL user interface (line since the boundaries are lines in our 2D axisymmetric domain).

The difference between the consumption rate of A, given by R_A, and the transport at the inlet and outlet of the reactor, given by $F_{A,0} - F_A$, is about $1 \cdot 10^{-7}$ (mol/s) with a relative tolerance of $1 \cdot 10^{-4}$. Although the discrepancy is small compared to the total species flow of 0.1 (mol/s), it still deserves an explanation.

Why is the discrepancy not of the magnitude of the machine precision, which would give a discrepancy in the 16th digit? There are several sources of errors:

- Quadrature error from the implementation of the FEM method
- Quadrature error from the geometry representation (very small since our geometry is rectangular)
- Quadrature error from the post evaluation of the integrals (also present for $F_{A,0}$, where the true solution is known)
- The algebraic error (controlled by tolerances)
- The truncation error for the FEM method

Let us start with the algebraic error. The nonlinear system of algebraic equations of the numerical model equations are solved in an approximate way using a so-called Newton method. This is an iterative method that starts with a guess for the solution **u** and then uses the gradient of **f** with respect to **u** to search for a new guess of **u**. This means that if we express the numerical model equations in the form:

$$\mathbf{f}(\mathbf{u}) = \bar{\mathbf{0}}$$

where we want to find the solution vector **u** that gives a zero function vector **f**, then we can only get close to zero.

The relative tolerance tells us that we will stop looking (stop iterating) for a new $\mathbf{u}$ when the relative changes in $\mathbf{u}$ are smaller than the tolerance. The absolute tolerance tells us how large changes we can accept in the concentration and temperature change before we terminate the search.

The residual vector tells us how close to zero we get in the equation above. The fact that we cannot solve the nonlinear system exactly may give us a contribution to the discrepancy that we obtain in the fluxes and reaction.

For example, if we increase the relative tolerance to a value of $1 \cdot 10^{-2}$ then the difference between the reaction rate and the flow of chemical species grows to $1 \cdot 10^{-5}$ (mol/s). If the discrepancy becomes too large, even with a small relative tolerance, then we may have to look into the residual vector for the solution of $\mathbf{f}(\mathbf{u})$ above; that is, investigate how each equation in the equation system deviates from 0. We may then choose to control the termination of the iterative solution with the residual and force the Newton solver to take a larger number of iterations.

The computation of the reaction rate and the transport rate, in combination with the tolerances and residual vector, tells us something about the accuracy in the solution of the numerical model equations.

We may find other derived values that we can use to check the solver tolerance and the termination condition, such as the conservation of energy in the system.

SENSITIVITY TO INPUT DATA

If a mathematical model is properly defined, it may also be well-posed. A mathematical model is well-posed if it has a unique solution that continuously depends on the problem data. If the model is not well-posed, this will be reflected in the numerical model and will cause problems in the solution process.

We can investigate the numerical solution's sensitivity to input by running a parametric sweep. If we look at the model equations, we can see that the reaction term has an exponential dependency of the activation energy through the Arrhenius equation, which could potentially be a nasty term. The figure below shows how the conversion along the length of the reactor varies with the activation energy. The profiles are smooth and continuous and if a surface plot is animated it even looks like an animation of an increasing reaction rate constant.

This tells us that the numerical solution seems to be well-behaved even with respect to this relatively nasty parameter in the Arrhenius expression. A similar plot is obtained for the normed temperature (T/T0), which also reveals the smoothness of the numerical solution with respect to the activation energy. A parametric sweep with respect to the heat of reaction shows a smaller influence and an even smoother behavior of the solution.

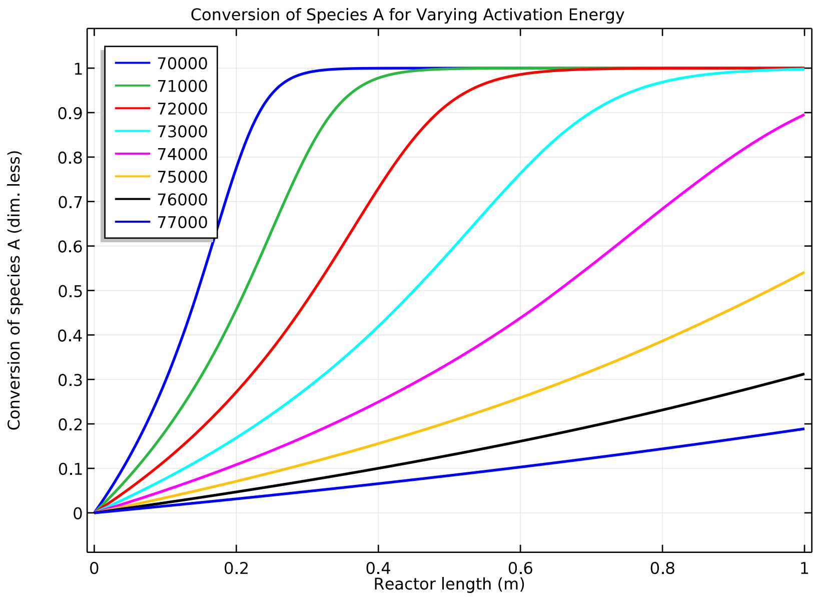


Figure 7: Conversion of species A for varying activation energy (J/mol).

The parametric sweeps with respect to critical model input may in this way increase our confidence to the numerical solution's ability to be an approximation of the mathematical solution.

COMPARISON WITH AN ANALYTICAL SOLUTION

The equation for the conservation of chemical species can be formulated for isothermal conditions and for plug flow which can be solved analytically. This represents a special case that allows us to compare the exact solution to the mathematical model and the numerical solution. Although not generally available, comparison with analytical solutions can be used both for laminar and turbulent flows, assuming that diffusion in the radial direction is small compared to the advective term in the axial direction in the bulk of the reactor.

When the radial dependency is removed from the mass transport equations, we obtain the following equation:

$$D_{\mathrm{eff}}\frac{\partial^2 c_A}{\partial z^2} + \left(-U\frac{\partial c_A}{\partial z}\right) - k_{\mathrm{f}} c_A = 0$$

Here, k_f denotes the rate constant (s^{-1}). The equation above has the following analytical solution:

$$c_A(z) = A_1 \exp(B_1 z) + A_2 \exp(B_2 z) .$$

From the boundary conditions, the constants A_1, A_2, B_1, and B_2 can be determined as

$$\begin{aligned} A_1 &= \frac{c_{A0} B_2 \exp(B_2 L)}{B_1 \exp(B_1 L) - B_2 \exp(B_2 L)} \\ A_2 &= c_{A0} - A_1 \\ B_1 &= \frac{U}{2D_{\mathrm{eff}}} - \sqrt{\left(\frac{U}{2D_{\mathrm{eff}}}\right)^2 - \frac{k_f}{D_{\mathrm{eff}}}} \\ B_2 &= \frac{U}{2D_{\mathrm{eff}}} + \sqrt{\left(\frac{U}{2D_{\mathrm{eff}}}\right)^2 - \frac{k_f}{D_{\mathrm{eff}}}} \end{aligned} ,$$

Where L is the length of the reactor. Using the same settings as for the numerical solutions above but now with a constant temperature of 332 K results in the plot in Figure 8 below.

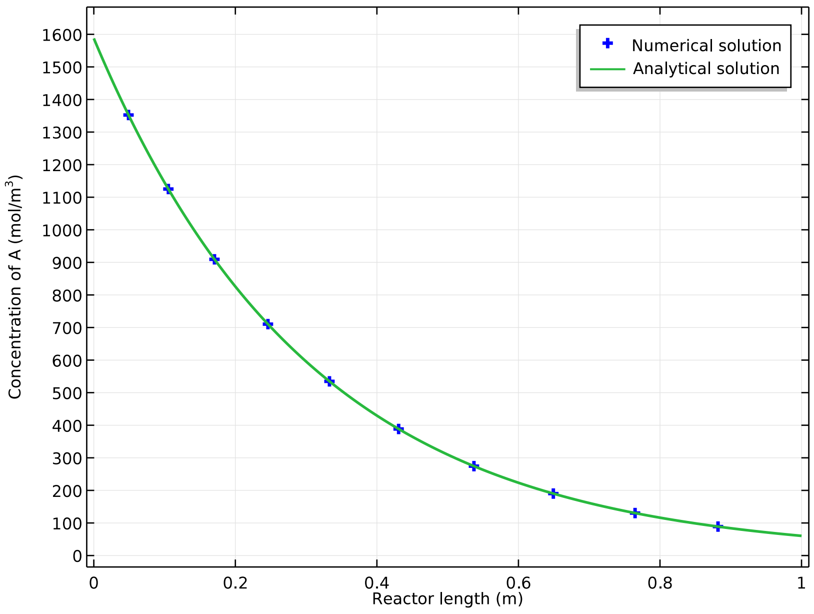


Figure 8: Concentration of species A along the length of the reactor computed using the numerical and analytical solution to the model equations.

As this plot shows, the agreement between the numerical and analytical solutions is very good. This implies that the numerical model is a good approximation of the mathematical model, at least for isothermal conditions.

Note again that the comparison with analytical solutions is not applicable in general. It can only be used for very special cases and the conclusions of such comparison must be treated with care.

ESTIMATING THE ACCURACY OF THE NUMERICAL MODEL

A brute-force way of estimating the accuracy in the numerical solution is to study the influence of discretization. In such a study, the full numerical model equations are solved for several different mesh resolutions and the results of relevant derived values are then compared for these mesh cases.

The background to such a study is that if the mathematical model is stable and has a unique solution, and in addition, the numerical model is consistent and stable, the numerical solution (u_{h}) should approach the true solution (u) when the mesh size h approaches 0. The truncation error may be defined as the difference between the numerical and the true solution to a problem:

$$\|u - u_h\| \propto O(h^p)$$

We can estimate the truncation error by plotting the difference in the numerical solution (u_{h}) for each mesh refinement. Also here, a relevant derived quantity of the numerical solution should be used, for example in our case the integral of the reaction rate over the reactor.

Such a procedure reveals if the numerical solution converges and also the convergence order (p). The plot of the logarithm should be able to give us the convergence order, provided that we have convergence:

$$\log\|u_{h,\,i+1} - u_{h,\,i}\| \propto p\log(h_{h,\,i+1})$$

If we plot the average relative change in the flux between mesh refinements in the tubular reactor model we get the plot in Figure 9.

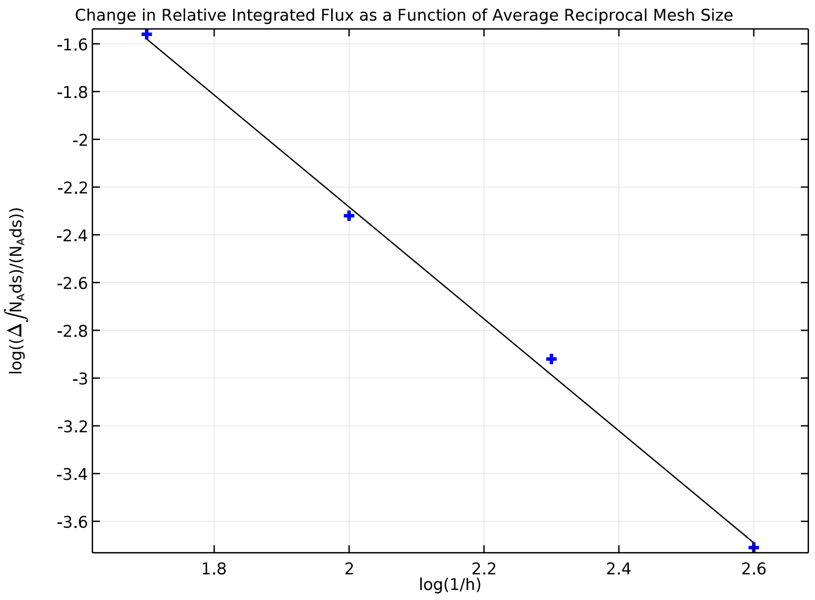


Figure 9: Relative change in flux between subsequent mesh refinements.

In this case, the mesh refinement is easy to define, since the mesh consists of rectangles. The size is therefore set to half of the previous mesh size for each refinement. The plot also hints that we are in the region where the numerical solution converges, since each mesh refinements leads to smaller changes in the solution. In the third refinement, the change in the flux is found in the third digit (a tenth of a percent) while in the fourth refinement, the change is found almost down in the fourth digit. The order of convergence is between 2 – 2.5 — that is, a reduction of the mesh size to one tenth gives more than two digits in improved accuracy.

Note that the convergence in numerical accuracy cannot take place forever. Eventually, we reach the limit of machine accuracy and the error may start to increase for each subsequent refinement due to rounding errors in the numerical calculations. For 2D and 3D simulations, this limit may be quite difficult to reach and may therefore be of small significance in practice.

CONCLUDING REMARKS

It may be difficult to verify the accuracy of a numerical model. For example, there may not be any relevant limits of the model that can be solved analytically like in our model above.

In addition, it may be difficult to create five mesh cases to obtain four mesh refinements, as the first mesh case may be difficult enough to solve with reasonable computer time and memory. However, it is always recommended to run sensitivity analyses of input data, mesh settings, and solver settings in order to investigate the stability of the mathematical and numerical models and if possible get an estimate of the accuracy of the numerical model.

Application Library path: `COMSOL_Multiphysics/Applications/tubular_reactor`
