

Dynamic Modeling & Transport Phenomena in Catalytic Fixed-Bed Reactors

Lucas van Dekken



Dynamic Modeling & Transport Phenomena in Catalytic Fixed-Bed Reactors

by

Lucas van Dekken

submitted in partial fulfillment of the requirements for the degree of
Bachelor of Science
in Double BSc Applied Physics & Applied Mathematics
at Delft University of Technology,
to be defended on February 2027.

Supervisors: Dr. Bijoy Bera
tud-tnw
Dr.ir. Z.J.G. (Zoë) Gromotka
tud-diam
Committee: Dr. N.V. (Neil) Budko
tud-diam
Dr. C.S. (Carlas) Smith
tud-3me

THIS THESIS IS CONFIDENTIAL AND CANNOT BE MADE PUBLIC.



Publication date:	September 17, 2026
Cover:	The cover image is a template demonstration background.
Copyright:	© 2026 Student Name

Optional custom colophon text below the publication table.

Made with [MyST](#) and [Typst](#) using the [JBoss Typst template](#).

Abstract

In this demo, we demonstrate how Jupyter Book can be used to create and publish a content rich paper that includes interactive elements such as code cells, visualizations, and multimedia. We will walk through the process of setting up a Jupyter Book, adding content, and deploying the final product online.

Keywords: Acetylene Semi-Hydrogenation, Heterogeneous Catalysis, Packed Bed Reactor, Transport Phenomena, Method of Lines, Urakawa Group, TU Delft

Contents

1. In plain language	1
1.1 About reactions	2
1.2 The reactor, physically	4
1.3 Parameters and provenance	5
2. Introduction	6
2.1 Background	6
3. Theory	7
3.1 1. The reaction network	7
3.2 2. Conservation laws	8
3.3 3. Boundary conditions	8
3.4 4. The four closures	9
3.5 5. Dimensionless groups	9
3.6 6. What the theory does not claim	9
4. Methods	10
4.1 1. Method of Lines	10
4.2 2. The radial operator and the centre node	10
4.3 3. The axial scheme	11
4.4 4. Volume weights	11
4.5 5. The integrator	11
4.6 6. Evaluation order	11
4.7 7. What is verified, and how	11
5. Results	13
6. Transient PBR Model — Acetylene Semi-Hydrogenation	16
6.1 1. Parameters	16
6.2 2. Discretisation & Indexing	17
6.3 3. Kinetic Laws	18
6.4 4. The ODE System (Method of Lines)	19
6.5 5. Execution & Implicit Time Integration	22
7. Conclusion	24
7.1 What was built	24
7.2 What was learned	24
7.3 What is not done	24
7.4 Next steps	25
8. Appendix	26
8.1 Cheatsheet	26
8.2 README	28
Bibliography	29

This page is intentionally left blank.

1. In plain language

We want to turn acetylene into ethylene — an important industrial reaction — without going one step too far and turning it into ethane.

The problem is that the same catalyst does both, and the second step is easy to overshoot. In practice the reaction is controlled by adjusting conditions: temperature, pressure, how long the gas spends in the reactor.

We are asking a different question: **can we control it better by varying the feed over time?** If the inlet concentration of the reactants pulses instead of holding steady, the catalyst surface never settles into one fixed state — and because the two reaction steps depend on the surface differently, a pulsing feed can favour the one we want.

To find out which pulses work, we first need a model. This work builds one: a description of a packed-bed reactor that tracks the gas along the tube, the gas inside the catalyst pellets, and the molecules sitting on the catalyst surface. It is built from parameters published in the literature rather than guessed, and every modelling simplification is written down next to the reason for it.

The model is not finished being tested against experiments, and the search for the best input profile has not started. What exists is the machinery to ask the question properly.

1.1 About reactions

The chemical reaction we're interested in is acetylene semi-hydrogenation to ethylene:



The problem is that the reaction doesn't stop at ethylene, but it also produces ethane:



We want to maximize the production of ethylene and minimize the production of ethane.

And keep the total production high, besides the selectivity.

The ideal catalyst would be one that selectively hydrogenates acetylene to ethylene and then stops.

In reality, we can control the selectivity by adjusting the reaction conditions (temperature, pressure, residence time, etc.). This is grounded in papers (the ones who talk about transients).

In this paper we're interested in manipulating the reaction through **transient concentration profiles** by pulsing the concentrations of reactants.

By modelling the reaction using a rate-equation model (grounded in the papers on microkinetic parameters) we can then use control theory to design optimal **Input Profiles** (in this case H_2 and C_2H_2 concentrations over time).

By finding the optimal input profile we can then use it to control the reaction in a lab-scale setup (plug flow reactor). This is grounded in another set of papers.

To achieve this, we first need a model of the reaction.

The reaction network consists of adsorption, surface reaction, and desorption steps (using $*$ to denote an adsorbed species, with vacant site fraction $\theta_* = 1 - \theta_H - \theta_{C_2H_2} - \theta_{C_2H_4}$):

1. Dissociative adsorption of H_2 :



$$R_1 = k_{1,ads} P_{H_2} \theta_*^2 - k_{1,des} \theta_H^2 \quad (1.4)$$

1. Molecular adsorption of C_2H_2 :



$$R_2 = k_{2,ads} P_{C_2H_2} \theta_* - k_{2,des} \theta_{C_2H_2} \quad (1.6)$$

1. Surface hydrogenation to ethylene (reversible):



$$R_3 = k_{3,fwd} \theta_{C_2H_2} \theta_H^2 - k_{3,rev} \theta_{C_2H_4} \theta_*^2 \quad (1.8)$$

1. Desorption of target product (ethylene):



$$R_4 = k_{4,des} \theta_{C_2H_4} - k_{4,ads} P_{C_2H_4} \theta_* \quad (1.10)$$

1. Over-hydrogenation to ethane (direct desorption to gas phase, irreversible):



$$R_5 = k_5 \theta_{C_2H_4} \theta_H^2 \quad (1.12)$$

Anyhow

We have the reaction. But it can only react on the catalyst surface, not in the gas phase.

So for our model we need to consider the gas-phase species, the adsorbed species and the catalyst surface as a whole.

We do this through the conservation laws. Paper 2008 Berger (1D dynamic axial model): \

$$\frac{\partial C_i}{\partial t} = -\frac{u_s}{\varepsilon_b} \frac{\partial C_i}{\partial z} + D_{\text{ax}} \frac{\partial^2 C_i}{\partial z^2} + \frac{\rho_s(1 - \varepsilon_b)C_t}{\varepsilon_b} R_i \quad (1.13)$$

It has accumulation ($\partial C_i / \partial t$), convective transport ($-\frac{u_s}{\varepsilon_b} \frac{\partial C_i}{\partial z}$), axial dispersion ($D_{\text{ax}} \frac{\partial^2 C_i}{\partial z^2}$, negligible under plug-flow criteria $Pe_{\text{ax}} > 100$), and catalytic surface generation/consumption (R_i).

□

1.2 The reactor, physically

A fixed-bed reactor is a tube packed with porous catalyst pellets. Gas flows through the voids between the pellets; the reaction happens on the internal surface of the pellets, which the gas must reach by diffusing into the pores.

Three length scales matter, and the model resolves all three:

Scale	What happens there	How the model treats it
Bed (cm–m)	Gas flows along the tube; concentration changes with position and time	Axial coordinate z , N_z cells
Pellet (mm)	Gas diffuses radially inward while reacting	Radial coordinate r , N_r nodes per cell
Surface (nm)	Molecules adsorb, react, and desorb on active sites	Fractional coverages θ_j

The separation of these scales is what makes the problem stiff: surface processes equilibrate in microseconds to milliseconds, while the gas composition along the bed evolves over seconds.

1.2.1 Why periodic operation is interesting

Under steady feed, the catalyst settles into a fixed surface composition. The selectivity to ethylene is then set by the balance between the desired partial hydrogenation and the undesired full hydrogenation to ethane — a balance that cannot be improved by changing a constant temperature or pressure alone.

Forcing the inlet composition to vary in time breaks that equilibrium. Because the two reaction pathways have different dependencies on the surface hydrogen coverage, a periodic inlet can drive the surface through states that a steady feed never reaches. The open question is which input profiles do this best — and that question requires a dynamic model to answer, which is what this work provides.

1.3 Parameters and provenance

Every quantity the model consumes is declared once, as a typed schema, before anything uses it. Nothing downstream is allowed to carry a bare number that did not come out of one of those declarations. That is what makes the model auditable from the outside, and it is what lets the test suite walk the registry and fail on a unit nobody checked.

1.3.1 Why units are handled where they are

The sources disagree. One prints kPa, another bar, a third Pa. Rather than requiring every call site to remember which is which, each kinetic parameter set converts **on the way in**, and the conversion factor is part of that set's own declaration. A rate constant therefore cannot be used with the wrong pressure scale, because the conversion is not a call-site decision.

The model works in SI throughout: metres, seconds, kilograms, moles, kelvin, pascals, with partial pressures in pascals.

1.3.2 Why parameter sets are types

Each kinetic literature source is its own schema. Switching between them is a **type change**, not a set of magic numbers scattered through the code. That means:

- A parameter that exists in one source and not another cannot be silently borrowed.
- A missing value is a construction error, not a `None` that fails later inside an integrator.
- The units of a set travel with the set.

1.3.3 Provenance

Each parameter set records where it came from and what was converted. Where a source reports a composite rate constant — a product of elementary constants reported as one number — that is what is stored, and the model does not attempt to decompose it. The consequence is stated in the Theory chapter: the model cannot reconstruct elementary surface kinetics from these numbers, and it does not pretend to.

Where a source reports adsorption terms individually, coverages *can* be written down, and that set is the one that supports genuine transient surface dynamics.

1.3.4 What is deliberately not parameterised

Some quantities that appear in the equations are not fitted parameters at all: the gas constant, the molar mass of the active metal, and the Faraday constant are fixed physics, held separately from the modelling parameters so that a change in a physical constant cannot be mistaken for a change in a fitted value.

2. Introduction

Jupyter Book has been rebuild from ground up using the MyST engine (Jupyter et al. 2025). This allows to export content in multiple output formats including HTML, PDF and docx. In this paper we present an overview of the possibilities and demonstrate its working.

In an introduction. you often cite. Than can be done in various ways, either using a .bib file or directly using the doi.

cite with doi

- `[@doi:10.25080/hwcj9957]` resulting in (Jupyter et al. 2025)
- `@doi:10.25080/hwcj9957` resulting in Jupyter et al. (2025)

cite from bib-file

- `{cite:t}borodzinskiSelectiveHydrogenationEthyne2006``` resulting in Borodziński and Bond (2006)
- `{cite:p}borodzinskiSelectiveHydrogenationEthyne2006``` resulting in (Borodziński and Bond 2006)

2.1 Background

Jupyter Book has been rebuild with the intend to export content in multiple output formats including HTML, PDF and docx. [Figure 1](#) provides this idea.

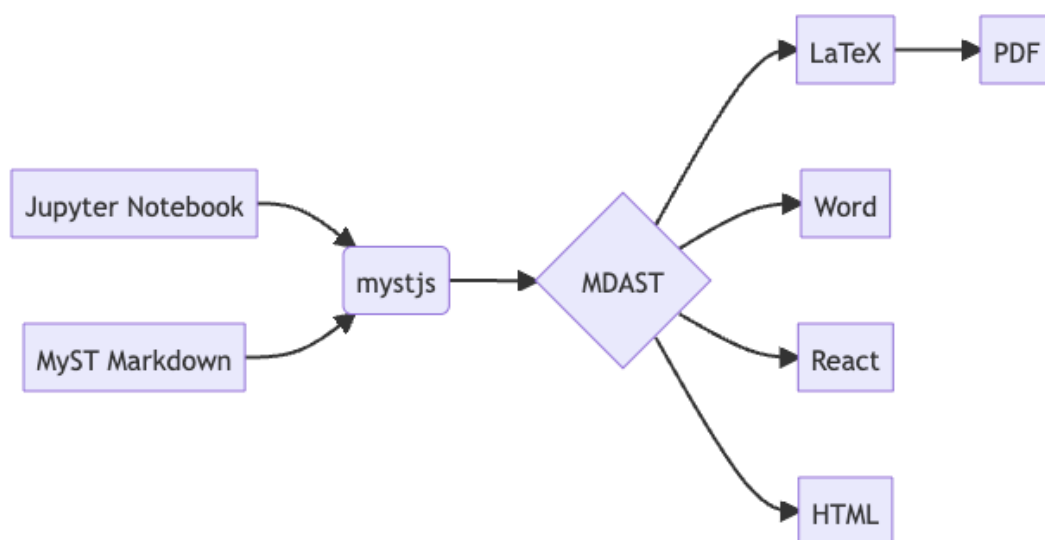


Figure 2.1: The myst engine allows Jupyter Notebook, markdown and even tex files to be converted to multiple output formats.

As exporting to different formats is possible, it is not always desired. Some content should only be visible in the HTML version, and some content only need to be included in the PDF version. You can use blocks like `+++ {"no-pdf": true}` to enable this, as shown below where the figure is seen in the HTML version but not in the PDF version.

Moreover, sometimes you want to have content [only showing up](#) in the pdf, if you use Typst you can use of a block `+++ {raw:typst}` and for LaTeX `+++ {raw:latex}`.

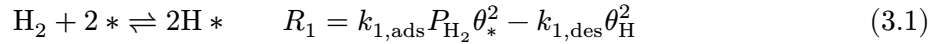
3. Theory

The model describes the transient behaviour of a fixed-bed catalytic reactor in which acetylene is selectively semi-hydrogenated to ethylene. This chapter states the physical and chemical formulation: the reaction network, the conservation laws, the boundary conditions, and the dimensionless groups that govern the regime.

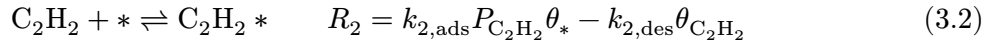
3.1 1. The reaction network

The chemistry is modelled with a Langmuir–Hinshelwood–Hougen–Watson (LHHW) mechanism. Five elementary steps compete for surface sites:

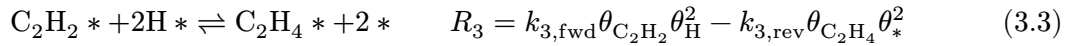
1. Dissociative adsorption of hydrogen



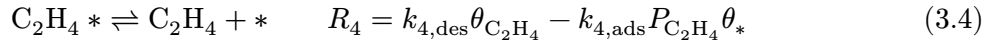
1. Molecular adsorption of acetylene



1. Surface hydrogenation to ethylene (reversible)



1. Desorption of the target product



1. Over-hydrogenation to ethane (irreversible, direct to gas phase)



The vacant-site fraction closes the site balance:

$$\theta_* = 1 - \theta_{\text{H}} - \theta_{\text{C}_2\text{H}_2} - \theta_{\text{C}_2\text{H}_4} \quad (3.6)$$

Why these five steps and not more. Ethane is deliberately absent from the adsorbed set. It leaves the surface immediately: there is no adsorbed ethane intermediate in the literature this model is parameterised from, and inventing one would be an unsourced mechanistic claim. The species set is chosen so that an element balance is checkable — the cheapest test that a stoichiometry table has been transcribed correctly.

Why macro rate laws. Every rate law in the model returns a rate directly from partial pressures and temperature, with no adsorbed intermediate appearing explicitly. That is what the available data supports: the parameter database holds *composite* rate constants — products of elementary constants that the source chose to report as a single number — so $d\theta/dt$ cannot be reconstructed from them. Refusing to invent the missing elementary constants costs mechanistic detail and buys a model whose every rate is traceable to a table in a published paper.

The one exception. The default kinetic set reports its adsorption terms individually rather than as products. Its coverages can therefore be written down, and with relaxation surface dynamics it becomes a genuine transient model — which is why it is the default. The other two parameterisations are algebraic-equilibrium only.

Non-monotonicity is preserved deliberately. All three laws are monotone in hydrogen and *decrease* in acetylene over part of their range — the rate maximum the sources both report. A model

in which the rate rises monotonically in its own reactant cannot exhibit the competitive-adsorption regime that makes periodic operation worth studying.

3.2 2. Conservation laws

The reactor is described by three coupled balances on a void-volume basis, following Berger et al. (2008) Eq. (1) and Eq. (2).

Bulk gas (along the bed, coordinate z):

$$\frac{\partial C_{b,i}}{\partial t} = -\frac{u_s}{\varepsilon_b} \frac{\partial C_{b,i}}{\partial z} + \frac{D_{ax}}{\varepsilon_b} \frac{\partial^2 C_{b,i}}{\partial z^2} + \frac{\rho_s(1 - \varepsilon_b)}{\varepsilon_b} \langle R_i \rangle_{\text{particle}} \quad (3.7)$$

Intra-particle gas (radially, coordinate r , geometry exponent s : 0 slab, 1 cylinder, 2 sphere):

$$\frac{\partial c_{p,i}}{\partial t} = \frac{D_e}{\varepsilon_p} \frac{1}{r^s} \frac{\partial}{\partial r} \left(r^s \frac{\partial c_{p,i}}{\partial r} \right) + \frac{\rho_s}{\varepsilon_p} R_i \quad (3.8)$$

Surface coverages (relaxation toward quasi-steady state, time constant τ):

$$\frac{\partial \theta_j}{\partial t} = \frac{\theta_j^{\text{ss}} - \theta_j}{\tau} \quad (3.9)$$

The bulk balance carries four terms: accumulation, convective transport, axial dispersion, and catalytic generation/consumption. Axial dispersion is negligible under the plug-flow criterion $Pe_{ax} > 100$, and is off by default.

Three conventions that had to be pinned down. Each is a silent factor of ε_p or ε_b if you get it wrong:

- u_s is the **superficial** velocity, based on the empty tube cross-section. The interstitial velocity is u_s/ε_b , and interstitial gas is what advection transports — so the transport term carries u_s/ε_b . Writing a bare u_s against a bulk-volume concentration is the same physics with the void fraction folded into the concentration; the two forms cannot be mixed without introducing exactly one spurious ε_b .
- D_e is **porosity-inclusive**, $D_e = \varepsilon_p D_m/\tau$, matching the transport reference, and is therefore divided by ε_p in the pellet balance to give diffusivity per unit pore volume. Defined instead as D_m/τ , that division would have to disappear. Both conventions circulate; they differ by exactly ε_p .
- The rate law is normalised to mol/(kg_cat·s), so the geometric prefactor carries the density: ρ_s/ε_p converts it to mol/(m³_pore·s) inside the pellet, and $\rho_s(1 - \varepsilon_b)/\varepsilon_b$ to mol/(m³_void·s) in the bulk. The two are consistent by construction — the particle equation volume-averaged over the pellet and multiplied by the bed solid fraction reproduces the bulk term exactly.

3.3 3. Boundary conditions

With dispersion switched off, the axial problem is **hyperbolic**, and the outflow condition is zero-gradient rather than Danckwerts. This is a consequence of the plug-flow assumption, not an independent modelling choice: imposing a Danckwerts outflow on a hyperbolic problem over-specifies it.

At the pellet centre the radial operator is singular at $r = 0$ and the limit must be taken before discretising. With symmetry $f'(0) = 0$, l'Hôpital resolves the 0/0 to give a centre coefficient of $2(1 + s)$ rather than a tabulated constant — see the Methods chapter.

3.4 4. The four closures

A packed-bed model is a stack of approximations. Each is isolated as an object with a small interface, so switching it is a change of argument rather than an edit inside the right-hand side. The design goal is that the right-hand side reads as a transcription of the governing equations, with no branches in it.

Closure	Question it answers	Default here
ExternalTransport	Does gas cross a film to reach the pellet?	No — the measured Carberry number is $\sim 3\text{e-}4$ against a 0.05 threshold, so the film is inactive by a factor of 150
RadialTransport	Does concentration vary inside a pellet?	Yes, resolved — not a convenience approximation: it is the reason the pellet is resolved at all. Switching it off gives the well-mixed-pellet limit
SurfaceDynamics	Does the catalyst remember?	Algebraic equilibrium by default — it is what the steady-state literature is calibrated under, and it is also the assumption the transient study is testing. Relaxation is one argument away
AxialDispersion	Does the gas back-mix?	Off — a packed bed at these flow rates is plug flow

Reducing every closure to its simplest setting gives a plug-flow, well-mixed-pellet, memoryless-catalyst, purely convective reactor. Turning them all on gives a dispersed reactor with a resolved pellet and a relaxing surface. Same right-hand side either way.

3.5 5. Dimensionless groups

The regime is characterised by the Péclet number for axial dispersion ($Pe_{ax} = u_s L / D_{ax}$, plug flow above ~ 100), the Damköhler number comparing reaction rate to convective transport, and the particle-scale Thiele modulus comparing intrinsic reaction rate to intra-particle diffusion. The Carberry number, quoted above for the film-resistance decision, compares observed to maximum external mass-transfer rate.

3.6 6. What the theory does not claim

The model does not attempt to reproduce the surface mechanism from first principles — the composite rate constants forbid it. It does not include an adsorbed ethane intermediate. It does not treat the catalyst as deactivating. Each of these is a deliberate boundary on the claims the model can support, and each is stated here rather than left implicit.

4. Methods

The governing equations are continuous in space and time. This chapter states how they are turned into a finite system that can be integrated: the spatial discretisation, the treatment of the pellet centre, the choice of scheme, and the integrator.

The design principle throughout is that **the right-hand side should read as a transcription of the governing equations**, with no branches in it. Every approximation is an object passed in, not a conditional inside the derivative.

4.1 1. Method of Lines

The Method of Lines (MOL) discretises space and leaves time continuous. The partial differential equations become a large system of ordinary differential equations in t , one per grid node, which is then handed to a stiff integrator.

Two spatial coordinates are discretised:

- **Axial** — N_z cells along the bed.
- **Radial** — N_r nodes inside each pellet.

The state vector is therefore the bulk concentrations at every axial cell, the intra-particle concentrations at every radial node of every cell, and the surface coverages. With the default closures the axial problem stays hyperbolic and the pellet carries N_r states per cell.

Why MOL rather than a fully implicit space-time scheme. MOL lets the spatial discretisation and the time integration be reasoned about separately, and it means the stiff integrator sees an explicit right-hand side it can evaluate and differentiate itself. The alternative — assembling a large sparse implicit system by hand — buys nothing here and makes the physics harder to read off the code.

4.2 2. The radial operator and the centre node

The radial operator is written in **flux form**,

$$\frac{1}{r^s} \frac{\partial}{\partial r} \left(r^s \frac{\partial c}{\partial r} \right) \quad (4.1)$$

discretised across cell faces at $r_{i\pm 1/2}$. Flux form rather than the algebraically equivalent $\frac{\partial^2 c}{\partial r^2} + \frac{s}{r} \frac{\partial c}{\partial r}$ because flux form has **one stencil instead of two**, so the two terms cannot drift out of the discrete conservation that the continuous operator satisfies.

4.2.1 The centre is where this gets interesting

The operator is singular at $r = 0$, and the limit has to be taken *before* discretising. Doing it properly:

$$\frac{1}{r^s} \frac{\partial}{\partial r} (r^s f') \longrightarrow f'' + \frac{s}{r} f' \quad (4.2)$$

As $r \rightarrow 0$ with symmetry $f'(0) = 0$, the second term's 0/0 resolves against the first via l'Hôpital to $(s-1)f''(0)$, giving a total of $(1+s)f''(0)$. The second derivative at the centre comes from the ghost node at $r = -dr$, and symmetry says its value equals the node at $+dr$:

$$f''(0) = \frac{c_1 - 2c_0 + c_{-1}}{dr^2} = \frac{2(c_1 - c_0)}{dr^2} \quad (4.3)$$

so the full centre operator is

$$\frac{2(1+s)(c_1 - c_0)}{dr^2} \longrightarrow \frac{6(c_1 - c_0)}{dr^2} \quad \text{for } s = 2 \quad (4.4)$$

The coefficient $2(1+s)$ is the one to carry, not the 6. Writing 6 hardcoded gives the right answer for spheres and silently the wrong one for cylinders (4) and slabs (2). Keeping the s in the expression costs nothing and makes the code correct for all three geometries, which is why it is *derived* rather than tabulated — and why there is a test asserting the coefficient is exactly $2(1+s)$.

4.3 3. The axial scheme

Backward-difference upwind is the default. Central differences would be second order and would **ring on the square-wave inlet** this model is meant to handle, producing concentrations below zero — and a stiff chemical source amplifies a negative concentration into a visible artefact immediately. Monotonicity beats order here.

TVD2 is available for smooth transients, where the extra order is worth the limiter. It is second order and monotone on a step, both of which are tested rather than asserted.

4.4 4. Volume weights

Node positions, face positions, and volume weights come from a single `RadialGrid` object, built once and handed to the operator. This is deliberate: a bug in the weights cannot silently disagree with a bug in the stencil if they come from the same object. The weights are exact for the volume element and sum to one — including for even node counts, which is a case that is easy to get wrong.

4.5 5. The integrator

The system is stiff, and the stiffness is physical rather than numerical: surface chemistry relaxes on a far shorter timescale than convective transport. An explicit integrator would be limited by the fastest surface timescale regardless of how slowly the bulk evolves.

The system is therefore integrated with an implicit multi-step / implicit Runge–Kutta solver from `scipy.integrate`, which can take steps set by accuracy rather than by the fastest mode. The Jacobian sparsity pattern is supplied: it is **block-tridiagonal in the axial direction**, which the integrator uses to avoid dense factorisation. That pattern is itself tested.

4.6 6. Evaluation order

The right-hand side evaluates in a fixed order — bulk, shell, particle, chemistry, then assembly — and the order matters. Each stage consumes the previous stage’s output, so reordering them silently changes the scheme from explicit to something else. It is stated in the code rather than left implicit for exactly that reason.

4.7 7. What is verified, and how

The test suite does not assert that the code runs; it asserts that the discretisation is *correct*. Representative checks:

- The first-order sphere and slab profiles match closed-form solutions.
- The sphere profile converges at **second order** under refinement.

- Volume weights are exact for the volume element and sum to one.
- The centre coefficient is exactly $2(1 + s)$.
- Upwind advection is exact for a constant and has the right sign and magnitude.
- TVD2 is second order and upwind is not; TVD2 is monotone on a step.
- Axial dispersion is exactly zero when switched off, and exact for a quadratic when on.
- The sparsity pattern matches the state vector and is block-tridiagonal.
- The particle equation, volume-averaged and multiplied by the bed solid fraction, reproduces the bulk source term — the consistency check that the two geometric prefactors agree.
- The right-hand side does not mutate its input.

The last two are the ones that catch real bugs: a mutating right-hand side breaks an integrator in ways that look like physics, and a mismatched prefactor produces a model that conserves nothing while still running.

5. Results

With JupyterBook it is possible to make your narrative and data-analysis available in a single file, through Jupyter Notebooks. It is even possible to create interactive materials using widgets on the website, but leave them out in the PDF version (using tags). To run the code below, click the



icon in the top right corner. This will connect with a kernel. If the kernel is loaded, click



to run the code.

Figure 1 shows the results of fitting our linear model to the data. Parameter a is optimal when the chi-squared curve is at its minimum, which corresponds to the smallest residuals and the best match between model and data.

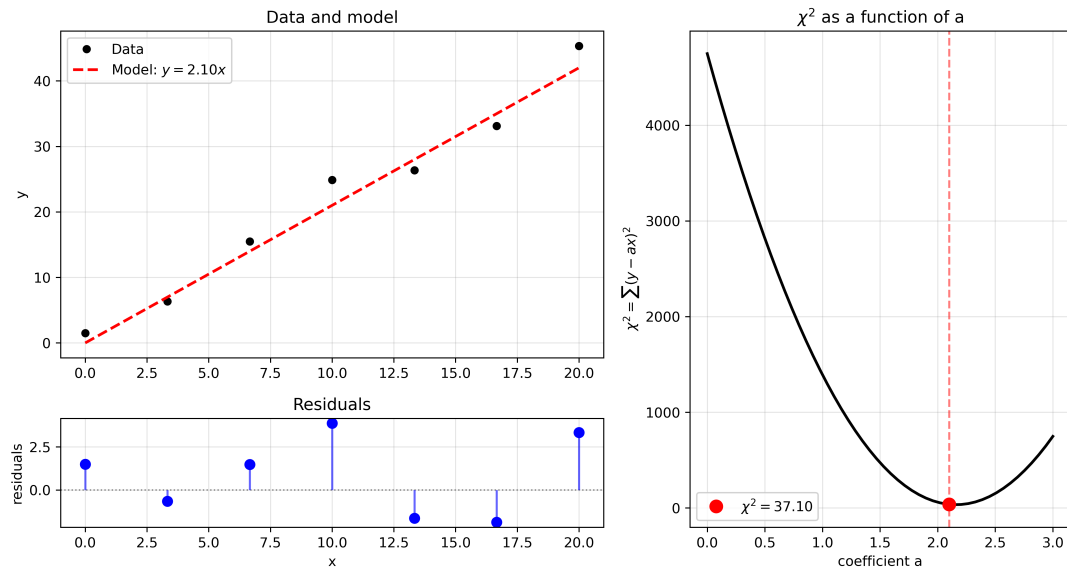


Figure 5.2: Fitting our data to a linear model with a single parameter a . The optimal value of a is found when the chi-squared curve (bottom) is at its minimum. The residuals (top right) are smallest at this point, and the model (top left) best matches the data.

6. Transient PBR Model — Acetylene Semi-Hydrogenation

Method-of-Lines packed-bed reactor (PBR) model for the transient semi-hydrogenation of acetylene (C_2H_2) to ethylene (C_2H_4) and ethane (C_2H_6) on a spherical catalyst pellet.

The model couples **bulk gas-phase advection** (plug flow), **intra-particle diffusion** (spherical coordinates), and **Langmuir–Hinshelwood surface kinetics** into a single system of 1 480 ODEs, integrated with an implicit BDF solver.

6.1 1. Parameters

All physical, operational, kinetic, and transport parameters are collected in a single `ModelParams` class. This keeps the model configuration in one place and makes it easy to swap parameter sets for different catalysts or operating conditions.

- **Reactor & hardware:** length L , pellet radius R_p , bed and pellet porosities (ϵ_b, ϵ_p), pellet density ρ_s , total active-site capacity C_t , superficial velocity u_s , and geometry factor s (2 = sphere).
- **Operating:** isothermal temperature T and the gas constant R_G . Inlet concentrations for all four gas species.
- **Kinetics:** Arrhenius reference rate constants k_{ref} and activation energies E_a for each elementary step (adsorption/desorption of H_2 , C_2H_2 , C_2H_4 , and the two surface reactions). Currently all set to 0.0 — placeholders pending literature values.
- **Transport:** effective diffusivity D_e for each gas species inside the pellet pores.

```
import numpy as np
from scipy.integrate import solve_ivp

#
=====
# 1. PARAMETERS (Open gelaten voor literatuurwaarden)
#
=====

class ModelParams:
    # --- Reactor & Hardware Parameters ---
    L = 0.5          # [m] Reactor lengte
    R_p = 0.002       # [m] Catalyst deeltjes radius
    epsilon_b = 0.4   # [-] Bed porositeit (void fraction)
    epsilon_p = 0.5   # [-] Deeltjes porositeit
    rho_s = 1200.0    # [kg/m3] Dichtheid van het katalysatordeeltje
    C_t = 1.0         # [mol/kg] Totale actieve site capaciteit
    u_s = 0.1         # [m/s] Superficial gas velocity
    s = 2            # [-] Geometrie factor (2 = sfeer)

    # --- Operationele Parameters ---
    T = 400.0         # [K] Isothermische temperatuur
    R_G = 8.314       # [J/mol.K] Gasconstante

    # Inlet concentraties [mol/m3]
    C_inlet = {
```

```

        'H2': 10.0,
        'C2H2': 1.0,
        'C2H4': 0.0,
        'C2H6': 0.0
    }

    # --- Kinetische Parameters (Arrhenius Reference & Activation) ---
    # Structuur: 'Reactie': (k_ref, E_a)
    # Let op: Dit zijn placeholders. Zowel adsorptie/desorptie als
    oppervlaktereacties.
    kinetics = {
        'ads_H2': (0.0, 0.0), # H2 + 2* ↔ 2H*
        'des_H2': (0.0, 0.0),
        'ads_C2H2': (0.0, 0.0), # C2H2 + * ↔ C2H2*
        'des_C2H2': (0.0, 0.0),
        'ads_C2H4': (0.0, 0.0), # C2H4 + * ↔ C2H4*
        'des_C2H4': (0.0, 0.0),
        'surf_R1': (0.0, 0.0), # C2H2* + 2H* → C2H4* + 2* (Reactie 1)
        'surf_R2': (0.0, 0.0) # C2H4* + 2H* → C2H6 + 3* (Reactie 2)
    }

    # --- Transport Parameters ---
    # Effectieve diffusiviteit in de poriën [m2/s]
    D_e = {
        'H2': 1e-6,
        'C2H2': 1e-6,
        'C2H4': 1e-6,
        'C2H6': 1e-6
    }

```

6.2 2. Discretisation & Indexing

The reactor is discretised into $N_z = 20$ axial nodes and each catalyst pellet into $N_r = 10$ radial nodes. The state vector is organised as three blocks:

1. **Bulk gas** C_b — shape (N_z , N_{gas}) — gas-phase concentrations in the bed void space.
2. **Pore gas** c_p — shape (N_z , N_r , N_{gas}) — gas concentrations inside the pellet pores.
3. **Surface coverage** θ — shape (N_z , N_r , N_{surf}) — fractional coverage of each surface species.

The total system dimension is $N_z \cdot N_{gas} + N_z \cdot N_r \cdot N_{gas} + N_z \cdot N_r \cdot N_{surf} = 80 + 800 + 600 = 1480$ ODEs.

```
#
```

```
=====
# 2. DISCRETISATIE & INDEXERING
```

```
#
```

```
=====
N_z = 20 # Aantal nodes over de reactorlengte
N_r = 10 # Aantal nodes over de deeltjesradius
```

```
# Species definitions en indexing voor arrays
gas_species = ['H2', 'C2H2', 'C2H4', 'C2H6']
surf_species = ['H_star', 'C2H2_star', 'C2H4_star']

N_gas = len(gas_species)
N_surf = len(surf_species)
```

6.3 3. Kinetic Laws

The reaction network follows a Langmuir–Hinshelwood mechanism with two surface reactions:

1. **R1:** $\text{C}_2\text{H}_2^* + 2\text{H}^* \xrightarrow{\text{r}} \text{C}_2\text{H}_4^* + 2^*$ (acetylene hydrogenation to adsorbed ethylene)
2. **R2:** $\text{C}_2\text{H}_4^* + 2\text{H}^* \xrightarrow{\text{r}} \text{C}_2\text{H}_6(\text{g}) + 3^*$ (ethylene hydrogenation to ethane, which desorbs directly)

Each elementary step has a forward and reverse rate. The net rates are computed from the pore-gas concentrations c_p and surface coverages θ at each (z, r) point. The site balance $\theta_v = 1 - \sum \theta_i$ gives the fraction of free sites.

The function returns two arrays: **R_gas** (net gas-phase production rates, mol/kg_cat/s) and **R_surf** (net surface accumulation rates, 1/s), which feed into the bulk and surface ODEs respectively.

```
#
=====
# 3. KINETISCHE WETTEN (De "Twee Reacties" C2H2 → C2H4 → C2H6)
#
=====

def calculate_rates(cp_gas, theta, p: ModelParams):
    """
    Berekent de netto productiesnelheden (R_i en R_j) voor een specifiek
    (z, r) punt.
    cp_gas: array van [H2, C2H2, C2H4, C2H6] in de porie
    theta: array van [H*, C2H2*, C2H4*]
    """
    # Unpack concentraties
    cp_H2, cp_C2H2, cp_C2H4, cp_C2H6 = cp_gas
    theta_H, theta_C2H2, theta_C2H4 = theta

    # Vrije sites (site balance)
    theta_v = 1.0 - (theta_H + theta_C2H2 + theta_C2H4)

    # Helper functie voor Arrhenius (Hier gesimplificeerd naar k_ref voor
    demonstratie)
    # In een volledig model: k = k_ref * exp(-Ea/R * (1/T - 1/Tref))
    k = lambda name: p.kinetics[name][0]

    # --- Elementaire stappen ---
    # 1. H2 Adsorptie/Desorptie (Dissociatief)
    r_ads_H2 = k('ads_H2') * cp_H2 * (theta_v ** 2) - k('des_H2') *
(theta_H ** 2)
```

```

# 2. C2H2 Adsorptie/Desorptie
r_ads_C2H2 = k('ads_C2H2') * cp_C2H2 * theta_v - k('des_C2H2') *
theta_C2H2

# 3. Oppervlaktereactie 1: C2H2* + 2H* → C2H4* + 2*
r_R1 = k('surf_R1') * theta_C2H2 * (theta_H ** 2)

# 4. C2H4 Desorptie/Adsorptie
r_des_C2H4 = k('des_C2H4') * theta_C2H4 - k('ads_C2H4') * cp_C2H4 *
theta_v

# 5. Oppervlaktereactie 2: C2H4* + 2H* → C2H6(g) + 3* (Ethaan
desorbeert direct)
r_R2 = k('surf_R2') * theta_C2H4 * (theta_H ** 2)

# --- Stoichiometrie: Netto Gas Productie (R_i) [mol / kg_cat s] ---
R_gas = np.zeros(N_gas)
R_gas[0] = -r_ads_H2          # H2
R_gas[1] = -r_ads_C2H2        # C2H2
R_gas[2] = r_des_C2H4          # C2H4
R_gas[3] = r_R2                # C2H6

# --- Stoichiometrie: Netto Oppervlakte Accumulatie (R_j) [1/s] ---
R_surf = np.zeros(N_surf)
R_surf[0] = 2 * r_ads_H2 - 2 * r_R1 - 2 * r_R2 # H*
R_surf[1] = r_ads_C2H2 - r_R1                  # C2H2*
R_surf[2] = r_R1 - r_des_C2H4 - r_R2           # C2H4*

return R_gas, R_surf

```

6.4 4. The ODE System (Method of Lines)

The `reactor_model` function is the right-hand side of the ODE system passed to `solve_ivp`. It evaluates the spatial derivatives from the outside (bulk) to the inside (pore arrow.r chemistry) at each time step.

6.4.1 Discretisation choices

- **Bulk advection** — *backward difference (upwind)*: $\partial C_b / \partial t = -(u_s / \varepsilon_b) (C_b[z] - C_b[z - 1]) / dz$. This is first-order accurate but unconditionally stable for the flow direction, which is essential for a convection-dominated PDE.
- **Intra-particle diffusion** — *central difference*: the spherical diffusion operator $\frac{1}{r^s} \frac{d}{dr} (r^s \frac{dc}{dr})$ is discretised with central differences at interior nodes. At the centre ($r = 0$), a symmetry condition replaces the singular term: the exact spherical limit is $2(1 + s)(c_1 - c_0) / dr^2$ (i.e. 6 for $s = 2$).
- **Pellet shell boundary** — the outermost radial node ($r = R_p$) is set equal to the bulk concentration (Dirichlet condition), representing the pseudo-homogeneous assumption that the pellet surface sees the bulk gas.
- **Particle-averaged rate** — the gas production integrated over the pellet volume is used as the source term in the bulk equation, coupling the two scales.

```

#
=====
# 4. DE DIFFERENTIAALVERGELIJKINGEN (MOL Framework)
#
=====

def reactor_model(t, y, p: ModelParams):
    """
    De core ODE functie die aan solve_ivp wordt gevoerd.
    Evalueert de ruimtelijke afgeleiden van Buiten (Bulk) naar Binnen
    (Porie → Chemie).
    """
    # 1. State vector y uitpakken in leesbare 3D/2D arrays
    # y opbouw: [C_b (N_z*N_gas), c_p (N_z*N_r*N_gas), theta
    (N_z*N_r*N_surf)]
    idx1 = N_z * N_gas
    idx2 = idx1 + (N_z * N_r * N_gas)

    C_b = y[0:idx1].reshape((N_z, N_gas))
    c_p = y[idx1:idx2].reshape((N_z, N_r, N_gas))
    theta = y[idx2:].reshape((N_z, N_r, N_surf))

    # Arrays initialiseren voor de afgeleiden (d/dt)
    dCb_dt = np.zeros_like(C_b)
    dcp_dt = np.zeros_like(c_p)
    dTheta_dt = np.zeros_like(theta)

    dz = p.L / N_z
    dr = p.R_p / (N_r - 1) # N_r punten betekent N_r-1 intervallen

    D_e_array = np.array([p.D_e[s] for s in gas_species])

    # Inlet randvoorwaarde array
    Cb_in = np.array([p.C_inlet[s] for s in gas_species])

    # 2. Ruimtelijke Loop over de Reactor (z-as)
    for z in range(N_z):

        # Pseudo-homogene aanname: Oppervlakteconcentratie =
        Bulkconcentratie
        C_s = C_b[z, :]

        # Deeltjes randvoorwaarde: Buitenste schil van deeltje (r = R_p)
        gelijk aan C_s
        c_p[z, -1, :] = C_s
        dcp_dt[z, -1, :] = 0.0 # Wordt algebraïsch gedicteerd door de bulk,
        geen ODE

        R_gas_integraal = np.zeros(N_gas)

```

```

# 3. Ruimtelijke Loop over het Deeltje (r-as)
for r in range(N_r - 1): # Buitenste schil is al gedefinieerd
    r_radius = r * dr

    # Kinetiek en Stoichiometrie
    R_gas, R_surf = calculate_rates(c_p[z, r, :], theta[z, r, :],
p)

    # Oppervlakte ODE (Accumulatie van actieve sites)
    dTheta_dt[z, r, :] = R_surf

    # Porie diffusie PDE (Discretisatie via eindige differenties)
    reactie_term = (p.rho_s * p.C_t / p.epsilon_p) * R_gas

    if r == 0:
        # Centrum (r=0): L'Hôpital / Symmetrie randvoorwaarde
        # d2c/dr2 approx = 2 * (c[1] - c[0]) / dr^2 (voor sfeer is
het 6 ipv 2 in de limiet)
        diff_term = (D_e_array / p.epsilon_p) * ( (1 + p.s) *
(c_p[z, 1, :] - c_p[z, 0, :]) / (dr**2) )
        dcp_dt[z, r, :] = diff_term + reactie_term
    else:
        # Interne nodes (CENTRAL DIFFERENCE SCHEME voor diffusie)
        dc_dr_plus = (c_p[z, r+1, :] - c_p[z, r, :]) / dr
        dc_dr_min = (c_p[z, r, :] - c_p[z, r-1, :]) / dr

        # (1/r^s) d/dr (r^s * dc/dr)
        r_plus_s = (r_radius + dr/2)**p.s
        r_min_s = (r_radius - dr/2)**p.s

        diff_term = (D_e_array / (p.epsilon_p * (r_radius**p.s) *
dr)) * (r_plus_s * dc_dr_plus - r_min_s * dc_dr_min)
        dcp_dt[z, r, :] = diff_term + reactie_term

    # Integreer de gasproductie over het deeltjesvolume voor de
bulkvergelijking
    # dV = afhankelijk van geometrie. Voor een sfeer: 4 * pi * r^2
* dr
    volume_gewicht = (r_radius**p.s) * dr
    R_gas_integraal += R_gas * volume_gewicht

    # Normaliseer integraal (gemiddelde reactiesnelheid in het deeltje)
    R_gas_avg = R_gas_integraal / ( (p.R_p**(p.s + 1)) / (p.s + 1) )

# 4. Bulk gas PDE (Plug-flow met BACKWARD DIFFERENCE / UPWIND
advectie)
# Dit garandeert stabiliteit voor stromingsprofielen (kijkt alleen
upstream).
```

```

        transfer_term = (p.rho_s * (1 - p.epsilon_b) * p.C_t / p.epsilon_b)
* R_gas_avg

        if z == 0:
            advectie_term = - (p.u_s / p.epsilon_b) * (C_b[z, :] - Cb_in) /
dz
        else:
            advectie_term = - (p.u_s / p.epsilon_b) * (C_b[z, :] -
C_b[z-1, :]) / dz

        dCb_dt[z, :] = advectie_term + transfer_term

# 5. Afgeleiden plat slaan tot één 1D array voor de solver
return np.concatenate((dCb_dt.flatten(), dcp_dt.flatten(),
dTheta_dt.flatten()))

```

6.5 5. Execution & Implicit Time Integration

The system is integrated with `scipy.integrate.solve_ivp` using the **BDF (Backward Differentiation Formula)** method. BDF is an implicit multistep method that is well-suited for stiff systems — here, the chemistry (fast adsorption/desorption) and transport (slow diffusion) operate on very different timescales, making explicit methods like RK45 impractical due to the tiny time steps they would require.

The initial condition is a zero state (empty reactor), and the simulation runs for 10 seconds of physical time. The solver reports the number of time steps taken, which gives a sense of the stiffness ratio.

```

#
=====
# 5. EXECUTIE & IMPLICIETE TIJDSINTEGRATIE
#
=====
import time

params = ModelParams()

# Bereken totale dimensie van het systeem
totaal_dimensie = (N_z * N_gas) + (N_z * N_r * N_gas) + (N_z * N_r *
N_surf)
y0 = np.zeros(totaal_dimensie) # Begin met een lege reactor (alles 0)

print(f"Initialisatie compleet. Start simulatie met systeemgrootte:
{totaal_dimensie} ODEs.")

# Tijdsperiode voor de simulatie
t_span = (0, 10) # Simuleer de eerste 10 seconden

# Gebruik van de impliciete BDF (Backward Differentiation Formula) methode.
# Waarom impliciet? Omdat chemische reacties zeer stijf (stiff) zijn

```

```

vergeleken met diffusie.
# Een expliciete methode (zoals standaard RK45) zal vastlopen op minuscule
benodigde tijdstappen.
print("Start integratie via impliciete BDF solver (dit kan even duren
afhankelijk van kinetiek)...")
start_time = time.time()

solution = solve_ivp(
    reactor_model,
    t_span,
    y0,
    method='BDF', # IMPLICIETE METHODE
    args=(params,)
)

end_time = time.time()
if solution.success:
    print(f"Simulatie succesvol afgerond in {end_time - start_time:.2f}
seconden!")
    print(f"Aantal genomen tijdstappen door BDF: {len(solution.t)}")
else:
    print("Simulatie gefaald:", solution.message)

```

Initialisatie compleet. Start simulatie met systeemgrootte: 1480 ODEs.
Start integratie via impliciete BDF solver (dit kan even duren afhankelijk
van kinetiek)...

Simulatie succesvol afgerond in 66.96 seconden!
Aantal genomen tijdstappen door BDF: 83

7. Conclusion

7.1 What was built

A transient packed-bed reactor model for the selective semi-hydrogenation of acetylene to ethylene, implemented as a small, tested Python package. The model couples bulk gas transport along the bed, intra-particle diffusion in the radial direction, and a relaxing catalyst surface, with the reaction chemistry supplied by one of three literature parameterisations behind a single interface.

The package separates the governing equations from the approximations applied to them. Four closures — external film resistance, intra-particle diffusion, surface dynamics, and axial dispersion — are objects passed into the model rather than branches inside it. The right-hand side reads as a transcription of the equations, which is what makes the model auditable from the outside.

7.2 What was learned

The modelling decisions are the work. The most valuable output of this phase is not the code but the reasoning attached to it: why flux form rather than the equivalent two-term radial operator; why the centre coefficient is $2(1 + s)$ rather than a hardcoded 6; why monotonicity beats order on a square-wave inlet; why the default kinetic set is the one whose adsorption terms are reported individually. Each of those is recorded where it was made, in the module it affects, rather than in a design document that would drift.

Refusing to invent is a design constraint, not a limitation. The parameter literature reports composite rate constants, so the elementary constants needed for a first-principles surface mechanism are simply not available. The model refuses to synthesise them. The cost is mechanistic detail; the benefit is that every rate in the model traces to a table in a published paper.

The interruptions were informative. The work was started and repeatedly interrupted by bug reports. Each interruption surfaced something the tests had not yet covered — which is the argument for writing tests that assert *correctness of the discretisation* rather than merely that the code runs. The suite now checks second-order convergence, exact volume weights, the centre coefficient, monotonicity on a step, and the consistency between the particle and bulk source terms.

7.3 What is not done

- **No parameter estimation.** The recorded next step is to specify the reaction mechanism, stoichiometry matrix, kinetic rate laws, and literature-based operating and kinetic parameters, then simulate forward before any fitting is attempted.
- **No validation against experimental data.** The model is internally consistent — the tests check that — but it has not been compared against measured transient responses.
- **No optimal input-profile design.** The stated purpose is to use the model to design transient inlet profiles that improve ethylene selectivity. That control step has not begun.
- **Single pellet geometry per run.** The operator supports slab, cylinder and sphere, but the parameter sets are configured for one geometry at a time.

7.4 Next steps

The immediate next step is to complete the parameter specification and run a forward simulation. Parameter estimation follows only once the forward problem is trustworthy.

8. Appendix

8.1 Cheatsheet

8.2 README

This is the TU Delft starterkit for open publishing with Jupyter Book. It provides a skeleton with basic settings to have a head start with your thesis / project.

It includes:

- a github deploy file taking care of deploying the website and building a pdf
- basic content files that can be altered to fit your project
- yaml files in the root folder to configure the book and the pdf build

8.2.1 🧑 Where to start

Already know about Jupyter Book, familiar with GitHub or GitLab, Markdown... start with cloning the [starterkit](#) and start writing working on your project. New to the ecosystem? Start with [the manual](#), and then move on to the starterkit.

8.2.2 🚩 Purpose

The purpose of this project and this manual is to enable others to use Jupyter Book for writing and publishing their own scientific and educational content. We hope to lower the technical barrier for researchers and educators to create and share their own resources, and to promote open science and open education practices.

8.2.3 📄 License

This book is licensed under the [Creative Commons Attribution-NonCommercial 4.0 International License](#), unless stated otherwise.

You are free to:

- Share — copy and redistribute the material in any medium or format
- Adapt — remix, transform, and build upon the material for any non-commercial purpose provided proper attribution is given.

8.2.4 ❌ Errors

If you find any errors in the content, please report them by opening an issue on the [GitHub repository](#).

8.2.5 💰 Funding

This project has received funding from the [Delft University of Technology Open Science Fund \(2025-2026\)](#).

8.2.6 ★ Identifier

TUDJBOSSTARTERKIT

Bibliography

- Berger, Rob J., Freek Kapteijn, Jacob A. Moulijn, et al. 2008. “Dynamic Methods for Catalytic Kinetics.” *Applied Catalysis A: General* 342 (1): 3–28. <https://doi.org/10.1016/j.apcata.2008.03.020>.
- Borodziński, Andrzej, and Geoffrey C. Bond. 2006. “Selective Hydrogenation of Ethyne in Ethene-Rich Streams on Palladium Catalysts. Part 1. Effect of Changes to the Catalyst During Reaction.” *Catalysis Reviews* 48 (2): 91–144. <https://doi.org/10.1080/01614940500364909>.
- Jupyter, Project, Evan Bolyen, J. Gregory Caporaso, et al. 2025. “Jupyter Book 2 and the MyST Document Stack: A Modular, Extensible, Web-Native Stack for Authoring and Publishing Computational Narratives.” In “Proceedings of the 24th Python in Science Conference.” Special issue, *Proceedings of the 24th Python in Science Conference*, SciPy, 173–93. <https://doi.org/10.25080/hwcj9957>.