Production-Function Approach

DRAFT

20 March 2020

# Concept

We separate the financial and conversion-efficiency aspects of the production process, which are generic across all technologies, from the physical and technical aspects, which are necessarily specific to the particular process. The motivation for this is that the financial and waste computations can be done uniformly for any technology (even for disparate ones such as PV cells and biofuels) and that different experts may be required to assess the cost, waste, and techno-physical aspects of technological progress.

# Formulation

## Sets

|  |  |  |
| --- | --- | --- |
| Set | Description | Examples |
|  | capital | equipment |
|  | fixed cost | rent, insurance |
|  | input | feedstock, labor |
|  | output | product, co-product, waste |
|  | metric | cost, jobs, carbon footprint, efficiency, lifetime |
|  | technical | temperature, pressure |

## Variables

|  |  |  |  |
| --- | --- | --- | --- |
| Variable | Type | Description | Units |
|  | calculated | unit cost | USD/unit |
|  | cost | capital cost | USD |
|  | cost | lifetime of capital | year |
|  | cost | scale of operation | unit/year |
|  | cost | fixed cost | USD/year |
|  | input | input quantity | input/unit |
|  | calculated | ideal input quantity | input/unit |
|  | waste | input efficiency | input/input |
|  | cost | input price | USD/input |
|  | calculated | ideal output quantity | output/unit |
|  | calculated | output quantity | output/unit |
|  | waste | output efficiency | output/output |
|  | cost | output price (+/-) | USD/output |
|  | calculated | metric | metric/unit |
|  | function | production function | output/unit |
|  | function | metric function | metric/unit |
|  | parameter | technical parameter | - |

## Cost

The per-unit cost is computed using a simple levelization formula:

## Waste

The waste relative to the idealized production process is captured by the parameters. Expert elicitation might estimate how the s would change in response to R&D investment.

* Waste of input: .
* Waste of output: .

## Production

The production function idealizes production by ignoring waste, but accounting for physical and technical processes (e.g., stoichiometry). This requires a technical model or a tabulation/fit of the results of technical modeling.

## Metrics

Metrics such as efficiency, lifetime, or carbon footprint are also compute based on the physical and technical characteristics of the process. This requires a technical model or a tabulation/fit of the results of technical modeling.

# Experts

Each expert elicitation takes the form of an assessment of the probability and range (10th to 90th percentile) of change in the cost or waste parameters or the production or metric functions.

# Implementation

Database tables (one per set) hold all of the variables and the expert assessments. These tables are augmented by concise code with mathematical representations of the production and metric functions.

The Monte-Carlo computations are amenable to fast tensor-based implementation in Python.

# Examples

## Idealized electrolysis of water

Here is a very simple model for electrolysis of water. We just have water, electricity, a catalyst, and some lab space. We choose the fundamental unit of operation to be moles of H2:

     H2O → H2 + ½ O2

Experts could assess how much R&D to increase the various efficiencies would cost. They could also suggest different catalysts, adding alkali, or replacing the process with PEM.

### Tracked quantities.

### Current design.

(due to mass transport loss on input)

(due to ohmic losses on input)

(due to mass transport loss on output)

(due to mass transport loss on output)

### Current costs.

(Al-Ni catalyst)

(effective lifetime of Al-Ni catalyst)

(rough estimate for a 50W setup)

### Current prices.

### Production function (à la Leontief)

### Metric function.

### Performance of current design.

(i.e., not profitable since it is positive)

# Implementation

See <<https://github.com/NREL/portfolio/tree/master/production-function/framework/code/tyche/>> for the tyche package that computes cost, production, and metrics from a technology design.

## Database tables

Each analysis case is represented by a Technology and a Scenario within that technology.

### Metadata about indices

The indices table simply describes the various indices available for the variables. The Offset column specifies the memory location in the argument for the production and metric functions.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Technology | Type | Index | Offset | Description | Notes |
| Simple electrolysis | Capital | Catalyst | 0 | Catalyst |  |
| Simple electrolysis | Fixed | Rent | 0 | Rent |  |
| Simple electrolysis | Input | Water | 0 | Water |  |
| Simple electrolysis | Input | Electricity | 1 | Electricity |  |
| Simple electrolysis | Output | Oxygen | 0 | Oxygen |  |
| Simple electrolysis | Output | Hydrogen | 1 | Hydrogen |  |
| Simple electrolysis | Metric | Jobs | 0 | Jobs |  |

### Design variables

The design table specifies the values of all of the variables in the mathematical formulation of the design.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Technology | Scenario | Variable | Index | Value | Units | Notes |
| Simple electrolysis | Base | Input | Water | 19.04 | g/mole |  |
| Simple electrolysis | Base | Input Efficiency | Water | 0.95 | 1 |  |
| Simple electrolysis | Base | Input | Electricity | 279 | kJ/mole |  |
| Simple electrolysis | Base | Input Efficiency | Electricity | 0.85 | 1 |  |
| Simple electrolysis | Base | Output Efficiency | Oxygen | 0.90 | 1 |  |
| Simple electrolysis | Base | Output Efficiency | Hydrogen | 0.90 | 1 |  |
| Simple electrolysis | Base | Capital cost | Catalyst | 0.63 | USD |  |
| Simple electrolysis | Base | Lifetime | Catalyst | 3 | yr |  |
| Simple electrolysis | Base | Fixed cost | Rent | 1000 | USD/yr |  |
| Simple electrolysis | Base | Scale |  | 6650 | mole/yr |  |
| Simple electrolysis | Base | Input price | Water | 4.8e-3 | USD/mole |  |
| Simple electrolysis | Base | Input price | Electricity | 3.33e-5 | USD/kJ |  |
| Simple electrolysis | Base | Output price | Oxygen | 3.0e-3 | USD/g |  |
| Simple electrolysis | Base | Output price | Hydrogen | 1.0e-2 | USD/g |  |

### Metadata for functions

The functions table simply documents which Python module and functions to use for the technology and scenario.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Technology | Module | Production | Metrics | Notes |
| Simple electrolysis | simple\_electrolysis | production | metrics |  |

### Parameters for functions

The parameters table contains ad-hoc parameters specific to the particular production and metrics functions. The Offset column specifies the memory location in the argument for the production and metric functions.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Technology | Scenario | Parameter | Offset | Value | Units | Notes |
| Simple electrolysis | Base | Oxygen production | 0 | 16.00 | g |  |
| Simple electrolysis | Base | Hydrogen production | 1 | 2.00 | g |  |
| Simple electrolysis | Base | Water consumption | 2 | 18.08 | g |  |
| Simple electrolysis | Base | Electricity consumption | 3 | 237 | kJ |  |
| Simple electrolysis | Base | Jobs | 4 | 1.5e-4 | job/mole |  |

### Units for results

The results table simply specifies the units for the results.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Technology | Variable | Index | Units | Notes |
| Simple electrolysis | Cost | Cost | USD/mole |  |
| Simple electrolysis | Output | Oxygen | g/mole |  |
| Simple electrolysis | Output | Hydrogen | g/mole |  |
| Simple electrolysis | Metric | Jobs | job/mole |  |

## Python module and functions

Each technology design requires a Python module with a production and metrics function.

# Simple electrolysis.  
  
  
# All of the computations must be vectorized, so use `numpy`.  
import numpy as np  
  
  
# Production function.  
def production(capital, fixed, input, parameter):  
  
 # Moles of input.  
 water = np.divide(input[0], parameter[2])  
 electricity = np.divide(input[1], parameter[3])  
  
 # Moles of output.  
 output = np.minimum(water, electricity)  
  
 # Grams of output.  
 oxygen = np.multiply(output, parameter[0])  
 hydrogen = np.multiply(output, parameter[1])  
  
 # Package results.  
 return np.vstack([oxygen, hydrogen])  
  
  
# Metrics function.  
def metrics(capital, fixed, input, outputs, parameter):  
  
 # Trivial jobs calculation.  
 jobs = parameter[4]  
  
 # Package results.  
 return np.vstack([jobs])