TU Delft Light

An Easy to Use LaTeX Template

TUDelft

This page intentionally left blank.

Delft University of Technology

[TU0000] LaTeX 101

TU Delft Light
An Easy to Use LaTeX Template

Supervisor:
Dr. Jan Jansen

Authors:
John Doe 0000001
Jane Doe 0000002

Abstract

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.
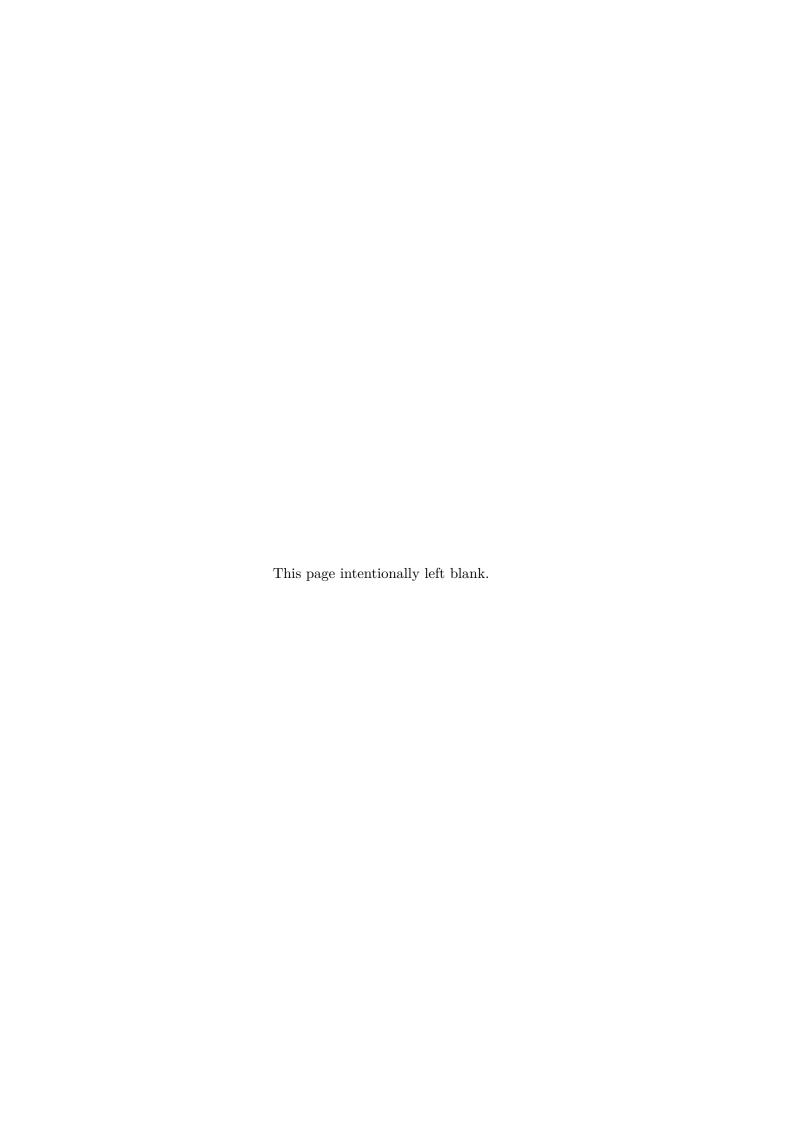
March 21, 2021

This page intentionally left blank.

Table of Contents

This page intentionally left blank.

List of Symbols

Abbreviations

ABCD   Ayy Bee See Dee

Roman Symbols

C_L Lift Coefficient $-$

V Velocity $\mathrm{kg\,m^{-1}}$

S Wing Area $\mathrm{m^2}$

Greek Symbols

ρ Density of Air $\mathrm{kg\,m^{-3}}$

This page intentionally left blank.

List of Figures

This page intentionally left blank.

List of Tables

This page intentionally left blank.

1 Example LaTeX Elements

This template has been developed for the [AE3200] Design Synthesis Exercise by Şan Kılkış and Munyung Kim. The source code can be modified and redistributed but the license file must remain intact. Refer to the `LICENSE.md` included with the template for details.

1.1 Tables & Figures

An example Table 1.1 and an example Figure 1.1 can be found in this section. When you label tables or figures, make sure to use 'tab:name' or 'fig:name', this is not necessary for syntax but makes organization and look-up of labels easier. For inserting 2+ figures in a row, look at the formatting of Figure 1.2. Using the `cleveref` package negates the need for manually typing 'Table' or 'Figure'. The syntax is as follows, note that the 'tab' in 'tab:exampletable' is not necessary for `cref` and is purely for organizational reasons. However a ',' cannot be utilized as this is interpreted as a list.

```
\cref{tab:exampletable}
```

The Tables below use the package `tabularx` which adjusts column spacing automatically to fit the table within the margins of the page. The syntax is as follows where 'L' is for Left Aligned, 'C' for Centered, and 'R' is for Right Aligned:

```
\begin{tabularx}{\textwidth}{L C C C}
```

In order to keep up the same appearance for all tables use the commands `toprule`, `midrule`, `bottomrule`, and `hdashline` to create the horizontal lines. NO VERTICAL LINES ARE ALLOWED!

Table 1.1: Example Table

Component	Mass [kg]	Location [m]	Location [% MAC]
Wing	425.4	5.74	40.00
Main Landing Gear	243.1	5.82	45.00
Fuel System	80.74	5.91	50.00
Flight Control System	48.61	6.08	60.00
Hydraulics	4.660	6.08	60.00
Wing Group	**802.5**	**5.80**	**43.85**
Fuselage	265.2	5.74	40.00
Engine	409.4	1.64	-
Avionics	490.9	4.39	-
H. Tail	42.93	13.2	-
V. Tail	66.43	12.6	-
Nose Gear	54.58	2.50	-
Electrical	217.4	6.16	67.12
AC & Anti-Ice	215.7	6.16	67.12
Furnishings	241.5	6.16	67.12
Fuselage Group	**2004**	**5.01**	**-2.32**
OEW C.G.	**2806**	**5.24**	**10.88**

Table 1.2: Example Table II

m	$\Re\{\underline{\mathfrak{X}}(m)\}$	$-\Im\{\underline{\mathfrak{X}}(m)\}$	$\mathfrak{X}(m)$	$\frac{\mathfrak{X}(m)}{23}$	A_m	$\varphi(m)$ / °	φ_m / °
1	16.128	+8.872	16.128	1.402	1.373	-146.6	-137.6
2	3.442	-2.509	3.442	0.299	0.343	133.2	152.4
3	1.826	-0.363	1.826	0.159	0.119	168.5	-161.1
4	0.993	-0.429	0.993	0.086	0.08	25.6	90
5	1.29	+0.099	1.29	0.112	0.097	-175.6	-114.7
6	0.483	-0.183	0.483	0.042	0.063	22.3	122.5
7	0.766	-0.475	0.766	0.067	0.039	141.6	-122
8	0.624	+0.365	0.624	0.054	0.04	-35.7	90
9	0.641	-0.466	0.641	0.056	0.045	133.3	-106.3
10	0.45	+0.421	0.45	0.039	0.034	-69.4	110.9
11	0.598	-0.597	0.598	0.052	0.025	92.3	-109.3

Figure 1.1: TU Delft Logo Flame

(a) TU Delft Logo Flame   (b) TU Delft Logo Flame

Figure 1.2: Two Figures Side-by-Side

1.2 References & Citations

The `biblatex` package is used for references with the default 'numeric' style for in-text citations and references [1]. The references sorting style is set to 'none' meaning that the references are sorted by the order in which they appear in text. A sample file `samplerefs.bib` is included to help when dealing with different types of publications.

```
\cite{citationtag}
```

1.3 Equations & Nomenclature

When typesetting equations, you need to use a nomenclature code when you introduce a variable for the FIRST time, such that the variable is listed on the list of symbols. An example is given below by Equation 1.1. With the current implementation, duplicate nomenclature items are not automatically removed.

$$L = \frac{1}{2}\rho V^2 S \cdot C_L \tag{1.1}$$

The the list of symbols for the above equation were generated with the code below:

```
\nomenclature[A]{ABCD}{Ayy Bee See Dee}
\nomenclature[B]{$C_L$}{Lift Coefficient \nomunit{-}}
\nomenclature[B, 01]{$V$}{Velocity \nomunit{kg.m^{-1}}}
\nomenclature[B, 02]{$S$}{Wing Area \nomunit{m^{2}}}
\nomenclature[G]{$\rho$}{Density of Air \nomunit{kg.m^{-3}}}
```

1.4 Units and Numbers

To have uniform spacing and formatting of numbers and units the `siunitx` package can be used. The syntax for displaying a number with its corresponding unit as "5 kg" is as follows:

```
\SI{5}{\kilogram}
```

Formatting of a unit of measure as "kg" is as follows, pay close attention to the lower-case call to `\si`.

```
\si{\kilogram}
```

1.5 Research Questions

Research questions can be formatted using the `questions` environment as follows:

```
\begin{questions}
    \item \label{rq:meaning} What is the meaning of life?
    \begin{questions}
        \item What is the answer to the Ultimate Question of Life, the
        Universe, and Everything?
    \end{questions}
\end{questions}
```

This produces the following output:

RQ-1. What is the meaning of life?

    RQ-1.1. What is the answer to the Ultimate Question of Life, the Universe, and Everything?

Note that individual questions can be referenced using the declared label, for example by using `\cref{rq:meaning}`. Doing so would render the reference as Research Question 1.

1.6 Quotes

Quotes can be typset using the `quoting` environment as follows:

```
\begin{quoting}
    ``Uni is great, faculty is awesome!''
\end{quoting}
```

   "Uni is great, faculty is awesome!"

References

[1] Lots of Coffee and Caffiene. *LaTeX: A Lovely Typesetting Language.* No One Publishing House of Bravos, 2019.

This page intentionally left blank.

A MATLAB Code

A.1 Optimization Run Case [RunCase.m]

```matlab
1   % Copyright 2018 San Kilkis, Evert Bunschoten
2   %
3   % Licensed under the Apache License, Version 2.0 (the "License");
4   % you may not use this file except in compliance with the License.
5   % You may obtain a copy of the License at
6   %
7   %     http://www.apache.org/licenses/LICENSE-2.0
8   %
9   % Unless required by applicable law or agreed to in writing, software
10  % distributed under the License is distributed on an "AS IS" BASIS,
11  % WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
12  % See the License for the specific language governing permissions and
13  % limitations under the License.
14
15  classdef RunCase < handle
16      %RUNCASE Summary of this class goes here
17      %   Detailed explanation goes here
18
19      properties
20          aircraft;            % Aircraft instance with all parameters and vars
21          x;                   % DesignVector object for fmincon & ease of use
22          x_final;             % Optimized Design Vector
23          converged = false;   % True if fmincon stopped w/o errors
24          options;             % fmincon options struct
25      end
26
27      properties (SetAccess = private, GetAccess = public)
28          cache = struct()     % Cache of Results & Constraints
29          run_parallel         % Bool, True for machines with >= 4 cores
30          iter_counter = 0     % Counts the number of function calls
31          start_time           % datetime at the start of optimization
32          end_time             % datetime at the end of optimization
33          sim_time;            % Total Sim. Time at end of Optimization [s]
34      end
35
36      methods
37
38          function obj = RunCase(aircraft_name, options)
39              % Displaying welcome message
40              type data\log\header.txt; fprintf('\n')
41
42              % Constructing the specified aircraft
43              obj.aircraft = aircraft.Aircraft(aircraft_name);
44              obj.init_design_vector(); % Creating the design vector object
45
46              % Augmenting options w/ OutputFnc
47              options.OutputFcn = @obj.cache_optimValues;
48              obj.options = options;
49              obj.cache.results = []; % Results caching
50              obj.cache.fmincon = []; % Solver caching
51              obj.cache.const = [];   % Constraint caching
52              obj.cache.time = [];    % Log of analysis time
53          end
54
55          function init_design_vector(obj)
56              dv = @optimize.DesignVector;
57              ac = obj.aircraft;
```

```matlab
58                    obj.x = dv({'lambda_1', ac.lambda_1, 0, 1.25;...
59                              'lambda_2', ac.lambda_2, 0.94, 1.25;...
60                              'b', ac.b, 0.71, 1.06;...
61                              'c_r', ac.c_r, 0.68, 1.15;...
62                              'tau', ac.tau, 0.16, 2.5;...
63                              'A_root', ac.A_root', 0.5, 1.2;...
64                              'A_tip', ac.A_tip', 0.5, 1.2;...
65                              'beta_root', ac.beta_root, 0, 1.7;...
66                              'beta_kink', ac.beta_kink, -0.8, 3.2;...
67                              'beta_tip', ac.beta_tip, -3.6, 3.6;...
68                              % Get these values from first initial run
69                              'A_L', ac.A_L, -1.5, 1.5;...
70                              'A_M', ac.A_M, -1.5, 1.5;...
71                              'W_w', ac.W_w, 0.6, 1.0;...
72                              'W_f', ac.W_f, 0.6, 1.0;...
73                              'C_d_w', ac.C_d_w, 0.8, 1.0});
74           end
75
76       function optimize(obj)
77           obj.start_time = datetime(); tic;
78           n_cores = feature('numcores');
79
80           % Launching either in Parallel or Serial Execution
81           try
82               if n_cores >= 4
83                   parpool(4)
84                   obj.run_parallel = true;
85               end
86           catch
87               obj.run_parallel = false;
88               warning(['Parallel Processing Disabled ' ...
89                        'or not Installed on Machine. Optimization '...
90                        'will execute as a serial process!'])
91           end
92
93           [opt, ~] = fmincon(@obj.objective,...
94                             obj.x.vector, [], [], [], [],...
95                             obj.x.lb, obj.x.ub, @obj.constraints,...
96                             obj.options);
97
98           obj.sim_time = toc;
99           obj.x_final = opt;
100          obj.end_time = datetime();
101          obj.converged = true;
102          obj.shutdown();
103       end
104
105      function [c, ceq] = constraints(obj, x)
106          disp('Constraints')
107          res = obj.fetch_results(x);
108          Cons = optimize.Constraints(obj.aircraft, res, obj.x);
109          c = Cons.C_ineq; ceq = Cons.C_eq;
110
111          % Caching of constraints
112          if isempty(obj.cache.const)
113              obj.cache.const.c = c;
114              obj.cache.const.ceq = ceq;
115          else
116              obj.cache.const.c(end+1, :) = c;
117              obj.cache.const.ceq(end+1, :) = ceq;
118          end
119       end
120
```

```matlab
121         function fval = objective(obj, x)
122             disp('Access from objective')
123             res = obj.fetch_results(x);
124             fval = res.W_f/obj.x.W_f_0;
125         end
126
127         function res = fetch_results(obj, x)
128             if ~obj.x.isnew(x) && ~isempty(obj.cache.results)
129                 res = obj.cache.results(end);
130             else
131                 disp('I asked for new runs')
132                 obj.x.vector = x; % Updates design vector w/ fmincon value
133                 obj.aircraft.modify(obj.x);
134                 obj.iter_counter = obj.iter_counter + 1;
135                 % Running Analysis Blocks
136                 if obj.run_parallel
137                     tic;
138                     spmd
139                         if labindex == 1
140                             temp = obj.run_aerodynamics();
141                         elseif labindex == 2
142                             temp = obj.run_structures();
143                         elseif labindex == 3
144                             temp = obj.run_loads();
145                         elseif labindex == 4
146                             temp = obj.run_performance();
147                         end
148                     end
149                     t = toc;
150                     fprintf('Parallel Process took: %.5f [s]\n', t)
151                     res.C_dw = temp{1};
152                     res.Struc = temp{2};
153                     res.Loading = temp{3};
154                     res.W_f = temp{4};
155                 else
156                     tic;
157                     res.C_dw = obj.run_aerodynamics();
158                     res.Loading = obj.run_loads();
159                     res.Struc = obj.run_structures();
160                     res.W_f = obj.run_performance();
161                     t = toc;
162                 end
163
164                 if isempty(obj.cache.results)
165                     obj.cache.results = res;
166                     obj.cache.time = t;
167                 else
168                     obj.cache.results(end+1) = res;
169                     obj.cache.time(end+1) = t;
170                 end
171             end
172         end
173
174         function A = run_aerodynamics(obj)
175             try
176                 Aero = aerodynamics.Aerodynamics(obj.aircraft);
177                 A = Aero.C_d_w;
178             catch
179                 A.C_D_w = NaN;
180             end
181         end
182
183         function L = run_loads(obj)
```

```matlab
184             try
185                 Loads = loads.Loads(obj.aircraft);
186                 L.M_distr = Loads.M_distr;
187                 L.L_distr = Loads.L_distr;
188                 L.Y_coord = Loads.Y_coord;
189             catch
190                 L.M_distr = ones(length(obj.x.A_M),1) * NaN;
191                 L.L_distr = ones(length(obj.x.A_M),1) * NaN;
192                 L.Y_coord = NaN;
193             end
194         end
195
196         function S = run_structures(obj)
197             try
198                 Structures = structures.Structures(obj.aircraft);
199                 S.W_w = Structures.W_w;
200                 S.V_t = Structures.V_t;
201             catch
202                 S.W_w = NaN;
203                 S.V_t = NaN;
204             end
205         end
206
207         function P = run_performance(obj)
208             try
209                 perf = performance.Performance(obj.aircraft);
210                 P = perf.W_fuel;
211             catch
212                 P.W_fuel = NaN;
213             end
214         end
215
216         function stop = cache_optimValues(obj, x, optimValues, state)
217             stop = false;
218             o = optimValues;
219             switch state
220                 case 'init'
221                     % hold on
222                     obj.cache.fmincon = o;
223                     obj.cache.fmincon.x = x;
224                 case 'iter'
225                     % Concatenate current point and objective function
226                     % value with history. x must be a row vector
227                     history = obj.cache.fmincon;
228                     temp.fval = [history.fval, o.fval];
229                     temp.x = [history.x, x];
230
231                     % Gradient caching of fmincon
232                     temp.gradient = [history.gradient,...
233                                     o.gradient];
234
235                     % Optimality caching of fmincon
236                     temp.firstorderopt = [history.firstorderopt,...
237                                     o.firstorderopt];
238
239                     temp.iteration = [history.iteration, o.iteration];
240                     temp.funccount = [history.funccount, o.funccount];
241                     obj.cache.fmincon = temp;
242
243                 case 'done'
244                     % hold off
245                 otherwise
246             end
```

```matlab
247            end
248
249        function shutdown(obj)
250            if obj.run_parallel
251                % Shutting Down Parallel Pool
252                poolobj = gcp('nocreate');
253                delete(poolobj);
254            end
255            obj.end_time = datetime();
256            if isempty(obj.sim_time)
257                obj.sim_time = obj.end_time - obj.start_time;
258            end
259        end
260    end
261
262    methods (Static)
263        function obj = load_run(run_file)
264            filename = [pwd '\data\runs\' run_file '.mat'];
265            try
266                loaded_obj = load(filename, 'run_case');
267                obj = loaded_obj.run_case;
268            catch
269                error('Supplied file has no property: run_case')
270            end
271        end
272    end
273 end
```