



Summer Fellowship Report

On

Python code for different sections of IS 800:2007
and
Preparing DDCL for Truss connection

Submitted by

Rachana Gupta

Under the guidance of

Prof.Siddhartha Ghosh
Civil Engineering Department
IIT Bombay

Under the Mentorship of

Sourabh Das
Project Research Associate

July 13, 2019

Acknowledgment

I would like to thank FOSSEE for providing me a platform to work on something I am very interested in. I am thankful to everyone who thought of having and involved in selection process based on screening tasks. I am grateful to be a part of team which promotes open source software.

I thank all the Osdag members, who are wonderful mentors and great team. I thank Sourabh Das (Project Research Associate), Ajmal Babu MS (Project Research Associate), Danish Ansari (Project Research Assistant), Yash Lokhande (Project Research Assistant), Darshan Viswakarma (Project Research Associate), Anand Swaroop (Project Research Associate), Anjali Jatav (Project Research Assistant) Deepti Reddy(Co-intern) and whole team, who made us feel welcome and planned all the tasks meticulously during this period.

I am grateful that I got a chance to work under Prof. Sidharth Ghosh, who took time to mentor us and monitored individual contributions as well.

Contents

1	Introduction	4
1.1	FOSSEE Summer Research Fellowship	4
1.2	What is Osdag?	4
1.3	Who can use ?	5
2	Python Code of IS 800: 2007	6
2.1	Python Code for Section 6 IS 800: 2007	7
2.2	Python Code for Section 7 IS 800: 2007	7
2.3	Python Code for Section 8 IS 800: 2007	7
2.4	Python Code for Annex E IS 800: 2007	8
3	Sectional Properties of I- section	9
3.1	Creating class for sectional properties of I section . . .	9
4	Preparation of DDCL	10
4.1	DDCL for Bolted Truss connection	10
4.2	DDCL for Welded Truss connection	10
5	Reference	11
6	Conclusion	12
	Appendices	14
A	Python Code for Section 6 IS: 800 2007	15
B	Python Code for Section 7 IS: 800 2007	20
C	Python Code for Section 8 IS: 800 2007	29
D	Python Code for Annex E IS: 800 2007	49

E	Creating class for sectional properties of I section	61
F	DDCL for Bolted Truss Connection	63
G	DDCL for Welded Truss Connection	81

Chapter 1

Introduction

1.1 FOSSEE Summer Research Fellowship

FOSSEE project promotes the use of FOSS (Free/Libre and Open Source Software) tools to improve quality of education in our country. FOSSEE encourages the use of FOSS tools through various activities to ensure availability of competent free software equivalent to commercial (paid) softwares.

The [FOSSEE](#) project is a part of the National Mission on Education through Infrastructure and Communication Technology (ICT), Ministry of Human Resources and Development, Government of India.

Osdag is one such open source software which comes under the FOSSEE project. Osdag internship is provided through FOSSEE project. Any UG/PG/PhD holder can apply for this internship. And the selection will be based on a screening task.

1.2 What is Osdag?

Osdag is Free/Libre and Open Source Software being developed for design of steel structures. Its source code is written in Python, 3D CAD images are developed using PythonOCC. Github is used to ensure smooth workflow between different modules and team members. It is in a path where people from around the world would be able to contribute to its development. FOSSEE's "Share alike" policy would improve the standard of the software when the source code is further modified based on the industrial and educational needs across the country.

Design and Detailing Checklist (DDCL) for different connections, mem-

bers and structure designs is one of the important bi-products of this project. It would create a repository and design guide book for steel construction based on Indian Standard codes and best industry practices.

1.3 Who can use ?

Osdag is created both for educational purpose and industry professionals. As Osdag is currently funded by MHRD, Osdag team is developing software in such a way that it can be used by the students during their academics and to give them a better insight look in the subject.

Osdag can be used by anyone starting from novice to professionals. It's simple user interface makes it flexible and attractive than other software. Video tutorials are available to help get started. The video tutorials of Osdag can be accessed [here](#).

Chapter 2

Python Code of IS 800: 2007

We are creating library for functions of IS 800: 2007, so that it can be used directly for compiling code of various modules of Osdag in future. So, I converted sections of IS 800: 2007 to python codes.

For creating python codes google strings are used to explain variables and output, where 'Args' section is used for defining variables and 'Return' is for output for each functions of IS 800:2007, as shown in figure;

```
# cl 7.1.2.1 design compressive stress of axially loaded member
@staticmethod
def cl_7_1_2_1_design_compressive_stress(K_L,alpha,E,f_y,r,gamma_m0):
    """
    Calculation of design compressive stress
    Args:
        K_L - Effective length of compression member in mm
        alpha - Imperfection factor
        E - Young's Modulus of Elasticity in N/mm**2
        f_y - Yield Stress in N/mm**2
        r - radius of gyration in mm

    Return:
        f_cd - Design strength of compression member in N/mm**2

    Note:
        Reference:
        IS 800:2007, cl.7.1.2.1
    """

    f_cc = (math.pi ** 2 * E) / (K_L / r) ** 2 #euler buckling stress
    lambda_ = math.sqrt(f_y / f_cc) #non-dimensional slenderness ratio
    phi = 0.5 * (1 + alpha * (lambda_ - 0.2) + lambda_ **2)
    srf = 1 / (phi + math.sqrt(phi ** 2 - lambda_ ** 2)) #stress reduction factor,kai
    f_cd = min(f_y / gamma_m0 * srf, f_y / gamma_m0)
    return f_cd
```

Tables of each sections of IS 800:2007 are stored as dictionary in python code, as shown below;

```

# Table 5 Partial Safety Factors for Materials, gamma_m (dict)
class IS800_2007(object):
    cl_5_4_1_Table_5 = {"gamma_m0": {'yielding': 1.10, 'buckling': 1.10},
                        "gamma_m1": {'ultimate_stress': 1.25},
                        "gamma_mf": {'shop': 1.25, 'field': 1.25},
                        "gamma_mb": {'shop': 1.25, 'field': 1.25},
                        "gamma_mr": {'shop': 1.25, 'field': 1.25},
                        "gamma_mw": {'shop': 1.25, 'field': 1.50}
    }

```

2.1 Python Code for Section 6 IS 800: 2007

Tension members are linear members in which axial forces acts and cause elongation, how such members should be designed is given in Section 6 ('Design of Tension Members') of IS 800: 2007 and its functions are converted into python codes so that it can be directly used in future for the development of module for design of tension Member in Osdag.It's Python codes is attached vide [Appendix - A](#).

2.2 Python Code for Section 7 IS 800: 2007

Compression members are the members which can take axial compressive load. These members usually fail by flexure buckling and section 7 of IS 800: 2007 explains how such members should be designed to resist buckling failure. Python code for this is attached vide [Appendix - B](#)

2.3 Python Code for Section 8 IS 800: 2007

Members subjected to predominant bending must have adequate design strength to resist bending moment and shear force and how such member should be designed properly so that it can work efficiently in its design period is explained in section 8 ("Design of Members subjected to bending") of IS 800: 2007. Python code for this is attached vide [Appendix - C](#)

2.4 Python Code for Annex E IS 800: 2007

Factors affecting elastic critical moment and its calculation constitutes Annex E ("Elastic Lateral Torsional Buckling") of IS 800: 2007.

Python code for this is attached in vide [Appendix - D](#)

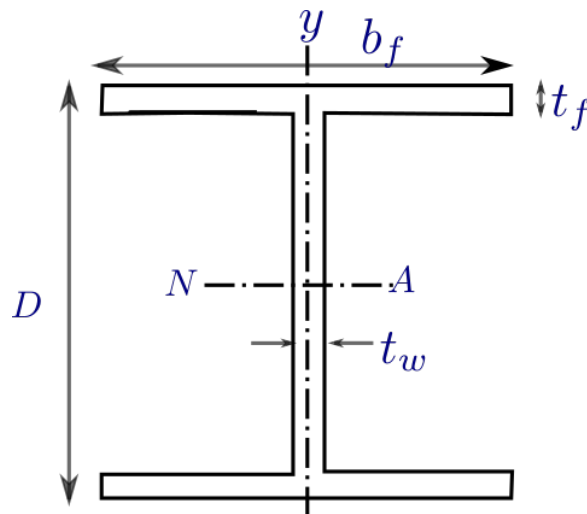
Git link for this is attached here: <https://github.com/Rachnagupta29/0sdag/pull/1>

Chapter 3

Sectional Properties of I- section

3.1 Creating class for sectional properties of I section

Class of sectional properties for I section is created so that Osdag software can also be used for the design of sections which are not in Steel Tables. And users will have freedom to design section according to their preference.



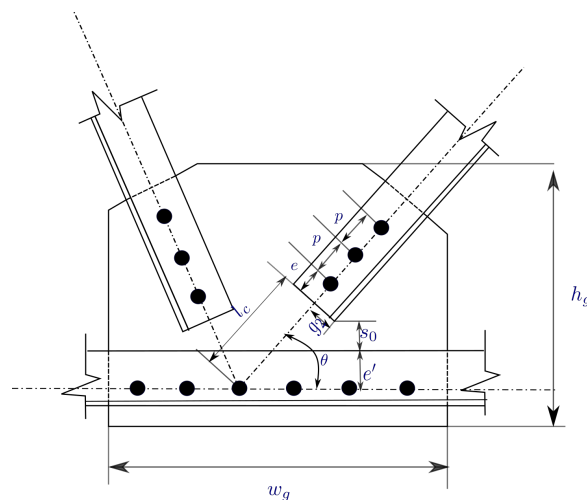
Python code for this is attached vide [Appendix - E](#)

Chapter 4

Preparation of DDCL

4.1 DDCL for Bolted Truss connection

I have made DDCL for Bolted Truss connection, which is going to be the design module of Osdag software in future. By this designers will be able to design efficient Bolted connection for truss for various geometrical arrangements.



The DDCL of Truss connection is attached vide [Appendix - F](#)

4.2 DDCL for Welded Truss connection

I prepared DDCL for weld connection, in which gusset plate is connected to members through welding in truss connection.

The DDCL of Truss connection is attached vide [Appendix -G](#)

Chapter 5

Reference

- IS 800:2007 Indian Standard Code of Practice for General Construction in Steel, 2007.
- IS 808:1989 Indian Standard Code of Practice for Dimensions or Hot Rolled Steel Beam, Column, Channel and Angle Sections, 1989.
- US Department of Transportation Federal Highway Administration, Guidelines for Design and Rating of Gusset-Plate Connections for Steel Truss Bridges.
- William A. et al, The Whitmore Section.
- Institute of Steel Development and Growth. Hand Book on Structural Steel Detailing, 2018a.
- N.Subramanian Design of Steel Structures. Oxford University Press, 13 edition, 2013.
- American Institute of Steel Construction. Steel Construction Manual, 13 edition, 2017.

Chapter 6

Conclusion

The internship enriched me with knowledge, skills and gave me a chance to interact with new people, which helped me to achieved several learning goals, and have moved a step further in achieving other. I got insight into how design is done in filed and the adverse factors that need to be considered. Internship has proved to be satisfactory and it has allowed as an opportunity to get an exposure of the practical implementation of theoretical fundamentals.

I learned technical things, like Osdag, Basics of Python, Latex, Git and Git hub

Throughout the internship, I found that several things are important:

- **Critical and Analytical Thinking**

To organize our tasks and assignment, we need to analyse our problems and assignment, and to formulate a good solution to the problem. We would have to set contingency plan for the solution, so that we are well prepared for the unforeseeable situations.

- **Time Management**

As overall project staff and programmer are always racing against tight time line and packed schedule, a proper time management will minimize facing overdue deadlines. An effective time management allows us to do our assignment efficiently and meet our schedules. Scheduling avoids time wastage and allows us to plan ahead, and gaining more as a result.

- **Goal Management**

It is better to sub-divide the goals to a few achievable tasks, so that we will be gaining more confidence by accomplishing those tasks.

- **Colleague Interactions**

In working environment, teamwork plays a vital role in contributing to a strong organization. Teamwork is also essential in reaching the goals of the organization as an entity. Thus, communicating and sharing is much needed in the working environment. Therefore, we should be respecting each other in work, and working together as a team, instead of working alone. This is because working together as a team is easier in reaching our targets, rather than operating individually.

I would like to once again appreciate everyone who has made my internship training a superb experience.

Appendices

Appendix A

Python Code for Section 6 IS: 800 2007

```
1 import math
2 """ SECTION 6 DESIGN OF TENSION MEMBERS """
3 # Table 5 Partial Safety Factors for Materials, gamma_m (dict)
4 class IS800_2007(object):
5     cl_5_4_1_Table_5 = {"gamma_m0": {'yielding': 1.10, 'buckling': 1.10},
6                          "gamma_m1": {'ultimate_stress': 1.25},
7                          "gamma_mf": {'shop': 1.25, 'field': 1.25},
8                          "gamma_mb": {'shop': 1.25, 'field': 1.25},
9                          "gamma_mr": {'shop': 1.25, 'field': 1.25},
10                         "gamma_mw": {'shop': 1.25, 'field': 1.50}
11                        }
12
13
14 # DESIGN OF TENSION MEMBER
15 # cl 6.1 Tension
16 @staticmethod
17 def cl_6_1_Design_strength_of_tesion_member(T_dg, T_dn, T_db):
18     """
19     Calculation of Design Strength of the member as per cl.6.1
20     Args:
21     T_dg - design strength due to yielding of gross section in N (float
22     ),
23     T_dn - design rupture strength of critical section in N (float),
24     T_db - design strength due to block shear in N (float)
25
26     Return:
27     T_d - design strength of the number under axial tension in N (
28     float),
29     Note:
30     References:
31     IS800:2007,cl.6.1
32     """
33     T_d = min(T_dg, T_dn, T_db)
34     return T_d
35
36 #cl.6.2 Design Strength due to yielding of Gross Section
37 @staticmethod
38 def cl_6_2_Design_strength_of_member_due_to_yielding_of_gross_section(A_g,
39 f_y):
40     """
41     Calculate the design strength due to yielding of gross section as
```



```

per cl.6.2
40  Args:
41     A_g- Gross area of cross section[in sq.mm](float)
42     f_y- Yield stress of the material in Mpa (float)
43
44  Return:
45     T_dg - Design strength of member due to yielding of gross section
in N(float)
46
47
48  Note:
49     Reference:
50     IS 800:2007, cl.6.2
51     """
52
53     gamma_m0 = IS800_2007.cl_5_4_1_Table_5["gamma_m0"]['yielding']
54     T_dg = A_g * f_y / gamma_m0
55     return T_dg
56
57
58 # cl.6.3 Design Strength Due to Rupture of Critical Section
59 # cl.6.3.1 Plates
60 @staticmethod
61 def cl_6_3_1_design_strength_in_tension(b, n, d_h, p_s, g, f_u, t):
62     """
63     Calculate the design strength in tension of a plate as per cl.6.3.1
64     Args:
65     A_n- net effective area of the member[in sq.mm](float)
66     f_u- ultimate stress of the material in Mpa(float)
67     b,t- width and thickness of the plate, respectively[in mm](float)
68     d_h- diameter of the bolt hole[2mm in addition to the diameter of
the hole,
69         in case the directly punched holes[in mm](float)
70     g- gauge length between the bolt holes[in mm](float list)
71     p_s- staggered pitch length between line of bolt holes[in mm](float
list)
72     n- number of bolt holes in the critical section (int)
73
74     Return:
75     T_dn - design strength in tension of a plate in N(float)
76
77     Note:
78     Reference:
79     IS 800:2007, cl.6.3.1
80     """
81
82     gamma_m1 = IS800_2007.cl_5_4_1_Table_5["gamma_m1"]['ultimate_stress']
83     sum_value = 0
84     for i in range(n - 1):
85         sum_value += (p_s[i] * p_s[i]) / (4 * g[i])
86
87     A_n = (b - n * d_h + sum_value) * t
88     T_dn = 0.9 * A_n * f_u / gamma_m1
89     return T_dn
90
91
92 # cl.6.3.2 Threaded Rods
93 @staticmethod
94 def cl_6_3_2_design_strength_of_threaded_rods_in_tension(A_n, f_u):
95     """
96     Calculate the design strength of threaded rods in tension as per cl

```

```

.6.3.2
97
98 Args:
99     A_n- net root area at the threaded section [in sq.mm](float)
100     f_u- ultimate stress of the material in Mpa(float)
101
102 Return:
103     T_dn - design strength of threaded rods in tension in N(float)
104
105 Note:
106     Reference:
107     IS800:2007, cl.6.3.2
108     """
109
110     gamma_m1 = IS800_2007.cl_5_4_1_Table_5["gamma_m1"]['ultimate_stress']
111     T_dn = 0.9 * A_n * f_u / gamma_m1
112     return T_dn
113
114
115 # cl.6.3.3 Single Angles
116 @staticmethod
117 def cl_6_3_3_design_strength_of_an_angle_connected_through_one_leg(A_nc ,
118     f_u, w, t, f_y, b_s, L_c, A_go):
119     """
120     Calculation of design strength of an angle connected through one
121     leg as per cl.6.3.3
122
123     Args:
124     w - outstanding leg width in mm(float)
125     b_s - shear lag width in mm(float)
126     L_c - length of the end connection, that is the distance between
127     the outermost
128     bolts in the end joint measured along the load direction or
129     length of the weld
130     along the load direction in mm(float)
131     A_nc - net area of the connected leg[in sq. mm](float)
132     A_go - gross area of the outstanding leg[in sq. mm](float)
133     t - thickness of the leg in mm(float)
134     f_u- ultimate strength of material in Mpa(float)
135
136 Return:
137     T_dn - design strength of an angle connected through onw leg in N(
138     float)
139
140 Note:
141     Reference:
142     IS800:2007, cl.6.3.3
143     """
144
145     gamma_m0 = IS800_2007.cl_5_4_1_Table_5["gamma_m0"]['yielding']
146     gamma_m1 = IS800_2007.cl_5_4_1_Table_5["gamma_m1"]['ultimate_stress']
147
148     X = max(0.7, f_u * gamma_m0 / f_y * gamma_m1)
149     beta = min(X, 1.4 - 0.076 * (w / t) * (f_y / f_u) * (b_s / L_c))
150     T_dn = (0.9 * A_nc * f_u / gamma_m1) / (beta * A_go * f_y / gamma_m0)
151     return T_dn
152
153 # cl 6.3.3 For preliminary sizing
154 @staticmethod

```

```

152 def cl_6_3_3_1_design_strength_of_net_section(n, A_n, f_u):
153     """
154     Calculation of rupture strength of net section for preliminary sizing
155     as per cl.6.3.3
156
157     Args:
158         A_n - net area of the total cross-section[in sq.mm](float)
159         f_u - ultimate tensile strength pf material in Mpa(float)
160         n - number of bolts (int)
161
162     Return:
163         T_dn - design strength of net section for preliminary sizing in N(
164         float)
165
166     Note:
167         Reference:
168         IS800:2007 cl.6.3.3
169     """
170     if n in [1, 2]:
171         alpha = 0.6
172     else:
173         if n == 3:
174             alpha = 0.7
175         else:
176             alpha = 0.8
177
178     gamma_m1 = IS800_2007.cl_5_4_1_Table_5["gamma_m1"]['ultimate_stress']
179     T_dn = alpha * A_n * f_u / gamma_m1
180     return T_dn
181
182 #cl.6.3.4 Other Section
183 #cl.6.4.1 Block shear strength of bolted connections
184 @staticmethod
185 def cl_6_4_1_block_shear_strength(A_vg, A_vn, A_tg, A_tn, f_u, f_y):
186     """
187     Calculation of the block shear strength of bolted connection as per
188     cl.6.4.1
189
190     Args:
191         A_vg - Minimum gross in shear along bolt line parallel to external
192         force [in sq.mm](float)
193         A_vn - Minimum net area in shear along bolt line parallel to
194         external force[in sq.mm](float)
195         A_tg - Minimum gross area in tension from the bolt hole to the toe
196         of the angle,
197         end bolt line, perpendicular to the line of force,
198         respectively [ in sq.mm](float)
199         A_tn - Minimum net area in tension from the bolt hole to the toe of
200         the angle,
201         end bolt line, perpendicular to the line of force,
202         respectively [in sq.mm]
203         f_u - Ultimate stress of the plate ,material in Mpa(float)
204         f_y - Yield stress of the plate material in Mpa(float)
205
206     Return:
207         T_db - block shear strength of bolted connection in N (float)
208
209     Note:
210         Reference:
211         IS800:2007, cl.6.4.1

```

```
204     """
205     gamma_m0 = IS800_2007.c1_5_4_1_Table_5["gamma_m0"]['yielding']
206     gamma_m1 = IS800_2007.c1_5_4_1_Table_5["gamma_m1"]['ultimate_stress']
207     T_db1 = A_vg * f_y / (math.sqrt(3) * gamma_m0) + 0.9 * A_tn * f_u /
208     gamma_m1
209
210     T_db2 = 0.9 * A_vn * f_u / (math.sqrt(3) * gamma_m1) + A_tg * f_y /
211     gamma_m0
212     T_db = min(T_db1, T_db2)
213
214     return T_db
214 # =====
```

Appendix B

Python Code for Section 7 IS: 800 2007

```
1 """ SECTION 7 DESIGN OF COMPRESSION MEMBERS """
2 import math
3 # Table 5 Partial Safety Factors for Materials, gamma_m (dict)
4 class IS800_2007(object):
5     cl_5_4_1_Table_5 = {"gamma_m0": {'yielding': 1.10, 'buckling': 1.10},
6                          "gamma_m1": {'ultimate_stress': 1.25},
7                          "gamma_mf": {'shop': 1.25, 'field': 1.25},
8                          "gamma_mb": {'shop': 1.25, 'field': 1.25},
9                          "gamma_mr": {'shop': 1.25, 'field': 1.25},
10                         "gamma_mw": {'shop': 1.25, 'field': 1.50}
11                        }
12
13
14 # cl 7.1 Design Strength
15 # cl.7.1.2
16 @staticmethod
17 def cl_7_1_2_design_copressive_strength_of_a_member(A_c, f_cd):
18     """
19     Calculation of design compressive strength
20     Args:
21     A_c - Effective sectional area (in square mm)
22     f_cd - Design Compressive stress(in N)
23
24     Return:
25     P_d - Design Compressive Strength of a member (in N)
26
27     Note:
28     References:
29     IS800:2007 cl.7.2
30     """
31
32     P_d = f_cd * A_c
33     return P_d
34
35
36 # cl 7.1.2.1 design compressive stress of axially loaded member
37 @staticmethod
38 def cl_7_1_2_1_design_compressive_stress(K_L,alpha,E,f_y,r,gamma_m0):
39     """
40     Calculation of design compressive stress
41     Args:
42     K_L - Effective length of compression member in mm
```

```

43         alpha - Imperfection factor
44         E - Young's Modulus of Elasticity in N/mm**2
45         f_y - Yield Stress in N/mm**2
46         r - radius of gyration in mm
47
48     Return:
49         f_cd - Design strength of compression member in N/mm**2
50
51     Note:
52     Reference:
53     IS 800:2007, cl.7.1.2.1
54     """
55
56     f_cc = (math.pi ** 2 * E) / (K_L / r) ** 2 #euler buckling stress
57     lambda_ = math.sqrt(f_y / f_cc) #non-dimensional slenderness ratio
58     phi = 0.5 * (1 + alpha * (lambda_ - 0.2) + lambda_ **2)
59     srf = 1 / (phi + math.sqrt(phi ** 2 - lambda_ ** 2)) #stress
reduction factor,kai
60     f_cd = min( f_y / gamma_m0 * srf, f_y / gamma_m0)
61     return f_cd
62
63
64     #cl 7.1.2.2 Calculation of buckling class of given cross-section
65     @staticmethod
66     def cl_7_1_2_2_Table_10_Buckling_class_of_cross_section(Cross_section,
t_f,t_w,h,b_f):
67         """
68         Defining Buckling Class of Cross-Section
69         Args:
70         Cross_section - Either 'Rolled_I_Section' or 'Welded_I_Section'
71         or 'Hot_rolled_hollow' or 'cold_Formed_hollow'
72         or 'Welded_Box_Section' or 'Channel,Angle,T,Solid Section' or '
Built_up_Member'
73
74         h- Depth of the section in mm
75         b_f - width of flange or width of section in case of welded box
section(mm)
76         t_f - Thickness of flange in mm
77         t_w - Thickness of web in mm
78
79     Return:
80     Dictionary of Buckling axis and Buckling class with Buckling
axis as key
81
82     Note:
83     Reference:
84     IS 800:2007, cl.7.1.2.2, Table_10
85     """
86     if Cross_section == "Rolled_I_Section":
87         if h / b_f > 1.2 :
88             if t_f <= 40:
89                 return {'z-z': 'a','y-y': 'b'}
90
91             if t_f>40 and t_f<=100:
92                 return {'z-z': 'b','y-y': 'c'}
93
94         if h / b_f <= 1.2:
95             if t_f <= 100:
96                 return {'z-z': 'b','y-y': 'c'}
97

```

```

98         if t_f>100:
99             return {'z-z': 'd','y-y': 'd'}
100
101     if Cross_section == "Welded_I_Section":
102         if t_f <= 40:
103             return {'z-z': 'b','y-y': 'c'}
104         if t_f>40:
105             return {'z-z': 'c','y-y': 'd'}
106
107     if Cross_section == "Hot_rolled_hollow":
108         return {'z-z': 'a','y-y': 'a'}
109
110     if Cross_section == "cold_Formed_hollow":
111         return {'z-z': 'b','y-y': 'b'}
112
113     if Cross_section == "Welded_Box_Section":
114         Buckling_Class_1 = 'b'
115         Buckling_Class_2 = 'b'
116
117         if b_f / t_f < 30:
118             Buckling_Class_1 = "c"
119
120         if h / t_w < 30:
121             Buckling_Class_2 = "c"
122
123         return {'z-z': Buckling_Class_1,'y-y': Buckling_Class_2}
124
125     if Cross_section == "Channel_Angle_T_Solid_Section" or
Cross_section == "Built_up_Member":
126         return {'z-z': 'c','y-y': 'c'}
127
128
129     #Imperfection Factor, alpha
130     # alpha for a given buckling class,'a','b','c' or 'd'
131     cl_7_1_Table_7_alpha = {
132         'a': 0.21,
133         'b': 0.34,
134         'c': 0.49,
135         'd': 0.76,
136     }
137
138     #Table 11 Effective Length of Prismatic Compression Members
139     @staticmethod
140     def cl_7_2_2_table11_effective_length_of_prismatic_compression_members(
L,BC=[]):
141
142         """
143         Effective length of Prismatic Compression Member when the
boundary conditions in the plane of buckling
144         can be assessed
145
146         Args:
147         BC - linked list of Boundary Conditions
148             =[BC_translation_end1,BC_rotation_end1,BC_translation_end2
,BC_rotation_end2]
149         L - Length of the Compression member in mm
150
151         Return:
152         K_L - Effective length of Compression Member in mm
153
154         Note:

```

```

155         Reference:
156         IS 800:2007, cl.7.2.2, Table_11
157         """
158
159         if BC == ['Restrained', 'Restrained', 'Free', 'Free'] or BC == ['
Restrained', 'Free', 'Free', 'Restrained']:K_L = 2.0 * L
160         elif BC == ['Restrained', 'Free', 'Restrained', 'Free']:K_L = L
161         elif BC == ['Restrained', 'Restrained', 'Free', 'Restrained']:K_L =
1.2 * L
162         elif BC == ['Restrained', 'Restrained', 'Restrained', 'Free']:K_L =
0.8 * L
163         elif BC == ['Restrained', 'Restrained', 'Restrained', 'Restrained']:
K_L = 0.65 * L
164         return K_L
165
166
167     # cl 7.1.2.1 design compressive stress of axially loaded member
168     @staticmethod
169     def design_compressive_stress(f_y, r ):
170
171         """
172         Calculation of design compressive stress
173         Args:
174             K_L - Effective length of compression member in mm
175             alpha - Imperfection factor
176             E - Young's Modulus of Elasticity in N/mm**2
177             f_y - Yield Stress in N/mm**2
178             r - radius of gyration of the section in mm
179
180         Return:
181             f_cd - Design strength of compression member in N/mm**2
182
183         Note:
184             Reference:
185             IS 800:2007, cl.7.1.2.1
186         """
187         Buckling_Class =
cl_7_1_2_2_Table_10_Buckling_class_of_cross_section(Cross_section, h,
b_f, t_f)
188         alpha = cl_7_1_Table_7_alpha["Buckling_Class"]["alpha"]
189         K_L = cl_7_2_2_effective_length_of_prismatic_Compression_members(
At_one_end_Translation, At_one_end_Rotation,
190
At_other_end_Translation,
191
At_other_end_Rotation, L)
192         f_cc = (pi * pi * E) / (K_L / r) ** 2
193         lambda_c = math.sqrt(f_y / f_cc)
194         phi = 0.5 * (1 + (alpha * (lambda_c - 0.2)) + (lambda_c * lamda_c))
195         srf = 1 / (phi + math.sqrt(phi ** 2 - lambda_c ** 2))
196         f_cd = min(((f_y / gamma_m0) * srf), f_y / gamma_m0)
197         return f_cd
198
199
200     # Design of Column Base
201     #cl.7.4.3 thickness of column base
202     @staticmethod
203     def cl_7_4_3_1_Calculation_of_thickness_of_column_base(w,a,b,t_f,f_y):
204
205         """
206         Calculation of thickenss of Column base

```



```

207     Args:
208         w - uniform pressure from below on the slab base in mm
209         a - Larger Projection in mm
210         b - Smaller Projection in mm
211         f_y - Yield Stress in N/mm**2
212         t_f - Thickness of flange of Compression member in mm
213
214     Return:
215         t_s - thickness of rectangular slab column base in mm
216
217     Note:
218         Reference:
219         IS 800:2007 cl.7.4.3.1,
220
221
222     """
223     gamma_m0 = IS800_2007.cl_5_4_1_Table_5["gamma_m0"]['yield_stress']
224     t_s = max(t_f, math.sqrt(2.5 * w * (a ** 2 - 0.3 * b ** 2) *
gamma_m0 / f_y))
225     return t_s
226
227     #cl.7.5.1.2 Loaded through one angle
228     #Table 12 - evaluation of constants K1,K2,K3 for effective slenderness
ratio
229     @staticmethod
230     def cl_7_5_1_2_table12_constant_K_1_K_2_K_3(
No_of_Bolts_at_Each_End_Connection, Connecting_member_Fixity):
231
232         """Value of constant K_1,K_2, K_3
233         Args:
234             No_of_Bolts_at_Each_End_Connection - Either more than 2 or 1,
235             Fixity - Either Fixed or Hinged.
236
237         Return:
238             [K_1,K_2,K_3]
239
240         Note:
241             Reference:
242             IS 800:2007 cl.7.5.1.2
243
244
245         """
246
247         if No_of_Bolts_at_Each_End_Connection >= 2:
248             if Connecting_member_Fixity == "Fixed":
249                 K_1 = 0.20
250                 K_2 = 0.35
251                 K_3 = 20
252
253             elif Connecting_member_Fixity == "Hinged":
254                 K_1 = 0.70
255                 K_2 = 0.60
256                 K_3 = 5
257
258         if No_of_Bolts_at_Each_End_Connection == 1:
259             if Connecting_member_Fixity == "Fixed":
260                 K_1 = 0.75
261                 K_2 = 0.35
262                 K_3 = 20
263
264             if Connecting_member_Fixity == "Hinged":

```

```

265         K_1 = 1.25
266         K_2 = 0.50
267         K_3 = 60
268
269         return [K_1,K_2,K_3]
270
271     #cl.7.5.1.2.Design strength of angle strut loaded through one leg
272     @staticmethod
273     def
274     cl_7_5_1_2_Calculation_of_design_strength_of_single_angle_strut_loaded_through_one_leg
275     (L, b_1, b_2, f_y, r_vv, t, E,K_list):
276         """
277         Calculation of design strength of single angle strut loaded
278         through one leg
279
280         Args:
281         L - Length of Angle section in mm
282         b_1,b_2 - width of legs of angle section in mm
283         f_y - yield stress in N/mm**2
284         r_vv - radius of gyration about minor axis in mm
285         t - thickness of the leg in mm
286         E - Young's Modulus of elasticity in N/mm**2
287         epsilon - yield stress ratio
288
289         Return:
290         f_cd - Design compressive strength of the section
291
292         Note:
293         Reference:
294         IS 800:2007 cl.7.5.1.2
295
296         """
297         [K_1,K_2,K_3] = K_list
298
299         alpha = 0.49 #according to ammendment 1
300
301         gamma_m0 = IS800_2007.cl_5_4_1_Table_5["gamma_m0"]['yielding']
302         epsilon = math.sqrt(250 / f_y)
303
304         lambda_vv = (L / r_vv) / (epsilon * math.sqrt(math.pi ** 2 * E /
305         250))
306
307         lambda_phi = (b_1 + b_2) / (2 * t * epsilon * math.sqrt(math.pi *
308         math.pi * E / 250))
309
310         lambda_e = math.sqrt(K_1 + (K_2 * lambda_vv ** 2) + (K_3 *
311         lambda_phi ** 2))
312
313         phi = 0.5 * (1 + alpha * (lambda_e - 0.2) + lambda_e ** 2)
314         f_cd = min(f_y / (gamma_m0 * (phi + math.sqrt(phi ** phi - lambda_e
315         ** 2))), f_y / gamma_m0)
316
317         return f_cd
318
319     #cl7.6 Laced column
320     #cl 7.6.1.5.Effective slenderness ratiion of lacing member
321     @staticmethod
322     def effective_slenderness_ratio_of_lacing_member(K_L, r_min):
323         """
324         Calculation of Effective slenderness ratio of lacing member
325         to account for shear deformation

```

```

319     Args:
320         K_L -effective length of column in mm
321         r_min - radius of gyration of column member in mm
322         SR_0 - actual maximum slenderness ration of column
323     Returns:
324         SR_eff - effective slenderness ratio of lacing
325     Note:
326         Reference:
327         IS 800:2007 cl 7.6.1.5
328     """
329
330     SR_0 = K_L/r_min
331     SR_eff = 1.05*SR_0
332     return SR_eff
333
334 #cl 7.6.2 Width of Lacing Bars
335 @staticmethod
336 def cl_7_6_2_width_of_lacing_bars(d):
337     """
338     Calculation of min width of Lacing Bars
339     Args:
340         d - nominal bolt/rivet diameter
341     Returns:
342         w_min - min Width of Lacing Bars
343     Note:
344         Reference:
345         IS 800:2007, cl 7.6.2
346     """
347
348     w_min = 3*d
349     return w_min
350
351 #cl 7.6.3 Thickness of Lacing Bars
352 @staticmethod
353 def cl_7_6_3_minimum_thickness_of_lacing_bars(lacing_type,L_eff):
354     """
355     Calculation of min Thickness of Lacing Bars
356     Args:
357         lacing_type - either 'single_lacing' or 'double_lacing'
358         L_eff - effective length of lacing bars
359     Returns:
360         t_min - minimum thickness of Lacing Bars
361     Note:
362         Reference:
363         IS 800:2007, cl 7.6.3
364     """
365     if lacing_type == 'single_lacing':
366         t_min = 1/40 *L_eff
367
368     else:
369         t_min = 1 / 60 * L_eff
370
371     return t_min
372
373 #Cl7.6.6 Design of lacing
374 #7.6.6.1 Transverse shear in the lacing bar
375 @staticmethod
376 def cl_7_6_6_1_transverse_shear_in_the_lacing_bar(P):
377     """
378     Calculation of Transverse shear in the lacing bar
379     Args:

```

```

380         P - axial load on column in N
381     Returns:
382         V_t_min - minimum design transverse shear in N
383     Note:
384         Reference:
385         IS 800:2007 cl 7.6.6.1
386     """
387     V_t_min = 2.5/100 * P
388
389     return V_t_min
390
391
392 # cl7.7 Batten plate
393 # cl7.7.1.4 effective slenderness ratio of batten plate
394 @staticmethod
395 def Cl_7_7_1_4_effective_slenderness_ratio_of_batten_plate(K_L,r_min):
396     """
397     Calculation of effective slenderness ratio
398     Args:
399         K_L -effective length of column in mm
400         r_min - minimum radius of gyration(r_x,r_y,r_z) of column
401 member in mm
402     Returns:
403         SR_eff - effective slenderness ratio of lacing
404     Note:
405         Reference:
406         IS 800:2007, cl 7.6.1.5
407     """
408     SR_eff = 1.1 * (K_L / r_min)
409     return SR_eff
410
411 # Design of Battens
412 # Battens
413 @staticmethod
414 def
415 cl_7_7_2_1_longitudinal_shear_transverse_shear_and_moment_at_connection
416 (P, S, C, N):
417     """
418     Calculation of longitudinal shear, transverse shear and moment
419 at connetion
420     Args:
421         P - total axial force on column in N
422         S - minimum transverse distance between the centroid of the
423 rivet/bolt
424         group/welding connecting the batten to the main member
425 in mm
426         N -number of parallel planes of battens
427         C - distance between centre -to- centre of battens in mm
428     Returns:
429         V_t - transverse shear force in N
430         V_b - longitudinal shear force along column axis in N
431         M - moment at connection in N*mm
432     Note:
433         Reference:
434         IS 800:2007, cl.7.7.2.1
435     """
436
437     V_t = 2.5 / 100 * P
438     V_b = V_t * C / (N * S)
439     M = V_t * C / (N * 2)

```

```
435     return (V_t, V_b, M)
436 #-----
```

Appendix C

Python Code for Section 8 IS: 800 2007

```
1 #
2 =====
3 import math
4 # Table 5 Partial Safety Factors for Materials, gamma_m (dict)
5 class IS800_2007(object):
6     cl_5_4_1_Table_5 = {"gamma_m0": {'yielding': 1.10, 'buckling': 1.10},
7                          "gamma_m1": {'ultimate_stress': 1.25},
8                          "gamma_mf": {'shop': 1.25, 'field': 1.25},
9                          "gamma_mb": {'shop': 1.25, 'field': 1.25},
10                         "gamma_mr": {'shop': 1.25, 'field': 1.25},
11                         "gamma_mw": {'shop': 1.25, 'field': 1.50}
12                        }
13
14     """ SECTION 8 DESIGN OF MEMBERS SUBJECTED TO BENDING """
15     # -----
16     # DESIGN OF MEMBER SUBJECTED TO BENDING
17
18     # cl 8.3.3 Effective length for cantilever Beam
19     @staticmethod
20     def cl_8_3_3_Table_16_Effective_length_for_cantilever_beam(L,
21     Restraint_Condition_1, Restraint_Condition_2,
22     Loading_condition):
23         """
24         Calculate effective length for cantilever beam of projecting
25         length L as per cl.8.3.3
26
27         Args:
28             L - Projecting Length of cantilever beam in mm (float)
29
30         Args:
31             L - Projecting Length of cantilever beam in mm (float)
32             D - Overall depth of the beam in mm (float)
33
34             Restrained_condition - Either "At support" or "At Top"
35             Restraint_Condition_1- "At support"
36             Restraint_Condition_2- "At Top"
37
38             At_support - Either "Continuous, with lateral restraint to
39             top"
40
41             or "Continuous, with partial torsional
```

```

37     restraint "
38         or "Continuous, with lateral and torsional
restraint "
39         or "Restrained laterally,torsionally and
40         against rotation on plan "
41         At_top - Either "Free"
42         or "Lateral restraint to top flange"
43         or "Torsional restraint"
44         or "Lateral and torsional restraint"
45         Loading_condition - Either "Normal" or "Destabilizing"
46
47     Returns:
48         L_LT =
49     cl_8_3_3_Table_16_Effective_length_for_cantilever_beam
50     Note:
51     References:
52     IS800:2007, Table 16 (cl 8.3.3)
53     """
54     if Restraint_Condition_1 == "Continuous, with lateral restraint to
top flange":
55         if Restraint_Condition_2 == "Free":
56             if Loading_condition == "Normal":
57                 return 3.0 * L
58             else:
59                 return 7.5 * L
60         if Restraint_Condition_2 == "Lateral restraint to top flange":
61             if Loading_condition == "Normal":
62                 return 2.7 * L
63             else:
64                 return 7.5 * L
65         if Restraint_Condition_2 == "Torsional restraint":
66             if Loading_condition == "Normal":
67                 return 2.4 * L
68             else:
69                 return 4.5 * L
70         if Restraint_Condition_2 == "Lateral and Torsional restraint":
71             if Loading_condition == "Normal":
72                 return 2.1 * L
73             else:
74                 return 3.6 * L
75     if Restraint_Condition_1 == "Continuous,with partial torsional
restraint":
76         if Restraint_Condition_2 == "Free":
77             if Loading_condition == "Normal":
78                 return 2.0 * L
79             else:
80                 return 5.0 * L
81         if Restraint_Condition_2 == "Lateral restraint to top flange":
82             if Loading_condition == "Normal":
83                 return 1.8 * L
84             else:
85                 return 5.0 * L
86         if Restraint_Condition_2 == "Torsional restraint":
87             if Loading_condition == "Normal":
88                 return 1.6 * L
89             else:
90                 return 3.0 * L
91     if Restraint_Condition_2 == "Lateral and Torsional restraint":

```

```

92         if Loading_condition == "Normal":
93             return 1.4 * L
94         else:
95             return 2.4 * L
96     if Restraint_Condition_1 == "Continuous,with lateral and torsional
restraint":
97         if Restraint_Condition_2 == "Free":
98             if Loading_condition == "Normal":
99                 return 1.0 * L
100            else:
101                return 2.5 * L
102        if Restraint_Condition_2 == "Lateral restraint to top flange":
103            if Loading_condition == "Normal":
104                return 0.9 * L
105            else:
106                return 2.5 * L
107        if Restraint_Condition_2 == "Torsional restraint":
108            if Loading_condition == "Normal":
109                return 0.8 * L
110            else:
111                return 1.5 * L
112        if Restraint_Condition_2 == "Lateral and Torsional restraint":
113            if Loading_condition == "Normal":
114                return 0.7 * L
115            else:
116                return 1.2 * L
117    if Restraint_Condition_1 == "Restrained laterally,torsionally and
against rotation on plan":
118        if Restraint_Condition_2 == "Free":
119            if Loading_condition == "Normal":
120                return 0.8 * L
121            else:
122                return 1.4 * L
123        if Restraint_Condition_2 == "Lateral restraint to top flange":
124            if Loading_condition == "Normal":
125                return 0.7 * L
126            else:
127                return 1.4 * L
128        if Restraint_Condition_2 == "Torsional restraint":
129            if Loading_condition == "Normal":
130                return 0.6 * L
131            else:
132                return 0.6 * L
133        if Restraint_Condition_2 == "Lateral and Torsional restraint":
134            if Loading_condition == "Normal":
135                return 0.5 * L
136            else:
137                return 0.5 * L
138
139
140    @staticmethod
141    def cl_8_3_1_Table_15_Effective_length_for_simply_supported_beams(L,D,
Restraint_Condition_1,
142    Restraint_Condition_2, Loading_Condition):
143        """
144        Calculate effective length against lateral torsional buckling
for simply supported Beams and girders
145        where no lateral restraint to the compression flange is
provided as per cl.8.3.1
146

```



```

147         Args:
148             L - Span of simply supported beams and girders in mm (
float)
149             D - Overall depth of he beam in mm (float)
150
151             Restraint_Condition - Either "Torsional Restraint" or "
warping Restraint"
152             Restraint_Condition_1- "Torsional Restraint"
153             Restraint_Condition_2- "Warping Restraint"
154
155             "Torsional Restrained" - Either "Fully restrained" or
156             "Partially restrained by bottom
flange support connection" or
157             "Partially restrained by bottom
flange bearing support"
158
159             "Warping_Restraint" - Either "Both flanges fully restrained
" or
160             "Both flanges partially restrained" or
161             "Compression flange fully restrained"
or
162             "Compression flange partially
restrained" or
163             "Warping not restrained in both flange
"
164
165
166             Loading_Condition - Either "Normal" or " Destabilizing"
167         Returns:
168         L_LT - c1_8_3_1_Effective length for simply supported
Beams in mm (float)
169         Note:
170             References:
171             IS800:2007, Table 15 (c1 8.3.1)
172     """
173
174     if Restraint_Condition_1 == "Fully Restrained":
175         if Restraint_Condition_2 == "Both flanges partially restrained"
:
176             if Loading_Condition == "Normal":
177                 return 0.70 * L
178             else:
179                 return 0.85 * L
180         if Restraint_Condition_2 == "Compression flange fully
Restrained":
181             if Loading_Condition == "Normal":
182                 return 0.75 * L
183             else:
184                 return 0.90 * L
185         if Restraint_Condition_2 == "Both flanges fully restrained":
186             if Loading_Condition == "Normal":
187                 return 0.80 * L
188             else:
189                 return 0.95 * L
190         if Restraint_Condition_2 == "Compression flange partially
Restrained":
191             if Loading_Condition == "Normal":
192                 return 0.85 * L
193             else:
194                 return 1.00 * L
195

```

```

196         if Restraint_Condition_2 == "Warping not restrained in both
197         flanges":
198             if Loading_Condition == "Normal":
199                 return 1.00 * L
200             else:
201                 return 1.20 * L
202         if Restraint_Condition_1 == "Partially restrained by bottom flange
203         support connection":
204             if Restraint_Condition_2 == "Warping not restrained in both
205             flanges":
206                 if Loading_Condition == "Normal":
207                     return 1.00 * L + 2 * D
208                 else:
209                     return 1.20 * L + 2 * D
210         if Restraint_Condition_1 == "Partially restrained by bottom flange
211         bearing support":
212             if Restraint_Condition_2 == "Warping not restrained in both
213             flanges":
214                 if Loading_Condition == "Normal":
215                     return 1.2 * L + 2 * D
216                 else:
217                     return 1.4 * L + 2 * D
218
219     @staticmethod
220     def cl_8_3_Effective_length_against_torsional_restraint(L, D, Beam_type
221     , Restraint_Condition_1,
222     Restraint_Condition_2,
223     Loading_Condition):
224         """
225         Calculation of effective length for given type of beam type as
226         per cl.8.3
227
228         Args:
229
230             L- Span of simply supported beams and girders in mm (float)
231         for
232             "
233         Simply_supported_with_no_lateral_restrained_to_the_compression_flanges
234         ",
235             Projecting Length of cantilever beam in mm (float) for
236             "Cantilever_beam",
237             Length of relevant segment between the lateral restraint
238         in mm (float) for
239             "Simply_supported_with_intermediate_lateral_restraints",
240             Centre-to-centre distance of the restraint member in mm (
241         float) for
242             'Beam provided with members to give effective lateral
243         restraint to compression flange at interval'
244
245             Beam_type - Either "
246         Simply_supported_with_no_lateral_restrained_to_the_compression_flanges"
247             or "
248         Simply_supported_with_intermediate_lateral_restraints"
249             or "
250         Beam_provided_with_members_to_give_effective_lateral_restrain_to_compression_flange_at_
251         "
252             or "Cantilever_beam"

```

```

238     FOR "
Simply_supported_with_no_lateral_restrained_to_the_compression_flanges"
239     Restraint_Condition - Either "Torsional Restraint" or "warping
Restraint"
240
241     Restraint_Condition - Either "Torsional Restraint" or "Warping
Restraint"
242     Restraint_Condition_1- "Torsional Restraint"
243     Restraint_Condition_2- "Warping Restraint"
244
245
246     "Torsional Restraint" - Either "Fully restrained" or
247     "Partially restrained by bottom
flange support connection" or
248     "Partially restrained by bottom
flange bearing support"
249
250     "Warping Restraint" - Either "Both flange fully restrained" or
251     "Compression flange fully restrained"
or
252     "Compression flange partially
restrained" or
253     "Warping not restrained in both flange
"
254
255
256     FOR "Cantilever_beam":
257
258     Restrained_condition - Either "At support" or "At Top" for "
Cantilever_beam"
259
260     Restraint_Condition_1- "At support"
261     Restraint_Condition_2- "At Top"
262
263     At_support - Either "Continuous, with lateral restraint to top
"
264     " or "Continuous, with partial torsional
restraint"
265     " or "Continuous, with lateral and torsional
restraint "
266     " or "Restrained laterally,torsionally and
against rotation on plan "
267
268     At_top - Either "free"
269     " or "lateral restraint to top flange"
270     " or "Torsional restraint"
271     " or "Lateral and torsional restraint"
272
273     Loading_condition - Either "Normal" or "Destabilizing"
274
275
276
277     Returns :
278     L_LT - effective_length_of_beam in mm (float)
279
280     Note:
281     References:
282     IS800:2007, cl 8.3.
283
284     ""
285

```

```

286     if Beam_type == "
Simply_supported_with_no_lateral_restrained_to_the_compression_flanges"
:
287         L_LT = IS800_2007.
cl_8_3_1_Table_15_Effective_length_for_simply_supported_beams(L, D,
Restraint_Condition_1,
288
Restraint_Condition_2,
289
Loading_Condition)
290     elif Beam_type == 'Simply supported with intermediate lateral
restraints':
291         L_LT = 1.2 * L
292
293     elif Beam_type == "
Beam_provided_with_members_to_give_effective_lateral_restrain_to_compression_flange_at_
":
294         L_LT = 1.2 * L #TODO:doubt-check
295     else:
296         L_LT = IS800_2007.
cl_8_3_3_Table_16_Efective_length_for_cantilever_beam(L,
Restraint_Condition_1,
297
Restraint_Condition_2, Loading_Condition)
298
299     return L_LT
300
301 # Design Strenth in Bending(Flexure)
302 @staticmethod
303 def cl_8_2_Design_strength_in_bending(M, M_d):
304     """ Calculation of design bending strength
305     Args:
306         M: Factored design moment in N*mm
307         M_d: design bending strength of the section in N*mm
308     Return:
309         M_d - design bending strength of the section in N*mm
310     Note:
311         References:
312             IS800:2007, cl 8.2.
313     """
314
315     return bool(M <= M_d)
316
317
318 @staticmethod
319 def cl_8_2_Design_bending_strength_of_laterally_unsupported_beam(z_p,
z_e, f_y, v, v_d, m_dv, plastic=False,
320
compact=False
):
321     """Calculation of bending strength of laterally unsupported beam
for low shear and high shear case
322     Args:
323         beta_b - 1 for plastic and compact
324                 Z_e/Z_p for semi-compact
325         z_e - Elastic section modulus of the cross section in mm**3
326         z_p - Plastic section modulus of the cross section in mm**3
327         f_y - yield stress of the material (in N/ mm**2 )
328         v - Factored design shear strength in N
329         v_d - Design shear strength in N
330         m_dv - Design bending strength under high shear as defined
in Cl 9.2 in N*m

```

```

331         plastic - True if beam is plastic else False
332         compact - True for compact section else False
333
334     Returns:
335         m_d - Design Bending strength in N*m
336     Note:
337     References:
338         IS800:2007, cl 8.2.1.2, cl. 8.2.1.3
339
340     """
341     beta_b = z_e / z_p #semi-compact section
342     if plastic is True:
343         beta_b = 1
344     if compact is True:
345         beta_b = 1
346     gamma_m0 = IS800_2007.cl_5_4_1_Table_5["gamma_m0"]['yielding']
347     if v <= 0.6 * v_d:
348         m_d = beta_b * z_p * f_y / gamma_m0
349
350
351     if v > 0.6 * v_d:
352         m_d = m_dv
353     return m_d
354
355     # cl8.2.2 DESIGN BENDING STRENGTH OF LATERALLY UNSUPPORTED BEAMS
356     # cl8.2.2.1 Elastic lateral torsional buckling moment
357     @staticmethod
358     def
359     cl_8_2_2_1_Elastic_lateral_torsional_buckling_moment_doubly_symmetric(
360     I_t, I_w, I_y, E, G, L_LT):
361     """
362     Calculation of elastic critical moment of lateral torsional
363     buckling for simply supported, prismatic members
364     with symmetric c/s
365     Args:
366
367         I_t - torsional constant
368         I_w - warping constant
369         I_y - moment of inertia about weaker axis
370         E - Young's Modulus of Elasticity
371         G - modulus of rigidity
372         L_LT - effective length for lateral torsional buckling
373     Return:
374         M_cr - Elastic lateral torsional buckling moment in N*mm
375     Notes:
376         Reference:
377         IS 800:2007, cl. 8.2.2.1.
378
379     """
380
381     M_cr = math.sqrt((math.pi ** 2 * E * I_y / L_LT ** 2) * (G * I_t +
382     math.pi ** 2 * E * I_w / L_LT ** 2))
383
384     return M_cr
385
386 @staticmethod
387 def cl_8_2_2_design_bending_strength_of_laterally_unsupported_beam(Z_p,
388 Z_e, L_LT, f_y, I_y, I_t, I_w, E, G, section, plastic=False, compact=
389 False):

```

```

386     """
387         Calculation of design bending strength of laterally
388         unsupported beam
389         Args:
390             Z_p - plastic section modulus with respect to extreme
391             compression fibre
392             Z_e - elastic section modulud with respect to extreme
393             compression fibre
394             L_LT - effective length for lateral torsional buckling
395             I_y - moment of inertia about minor axis of c/s
396             f_y - yield stress
397             I_t - torsional constant
398             I_w - warping constant
399             E - modulus of elasticity
400             G - modulus of rigidity
401             plastic - boolean True if section is plastic
402             compact - boolean True if section is compact
403             section - Either 'Rolled_steel_section' or "
404             Welded_steel_section'
405
406         Returns:
407             M_d - Design bending strength of laterally unsupported beam
408             in N*mm
409
410         Note:
411             Reference:
412             IS 800:2007, cl. 8.2.2.
413
414         """
415         beta_b = Z_p / Z_e
416         if plastic is True:
417             beta_b = 1
418         if compact is True:
419             beta_b = 1
420
421         if section == 'Rolled_steel_section':
422             alpha_LT = 0.21
423         elif section == 'Welded_steel_Connection':
424             alpha_LT = 0.49
425
426         M_cr = IS800_2007.
427         cl_8_2_2_1_Elastic_lateral_torsional_buckling_moment_doubly_symmetric(
428             I_t, I_w, I_y, E, G, L_LT)
429         f_cr_b = M_cr / (beta_b * Z_p)
430         Lambda_LT = min(math.sqrt(f_y / f_cr_b), math.sqrt(1.2 * Z_e * f_y
431             / M_cr))
432         phi_LT = 0.5 * (1 + alpha_LT * (Lambda_LT - 0.2) + Lambda_LT ** 2)
433         X_LT = min(1.0, 1 / (phi_LT + math.sqrt(phi_LT ** 2 - Lambda_LT **
434             2)))
435
436         if Lambda_LT < 0.4:
437             X_LT = 1
438
439         gamma_m0 = IS800_2007.cl_5_4_1_Table_5["gamma_m0"]['yielding']
440
441         f_bd = X_LT * f_y / gamma_m0
442
443         M_d = beta_b * Z_p * f_bd
444

```

```

438     return M_d
439
440
441 # cl8.4 Shear Design
442 @staticmethod
443 def cl_8_4_design_shear_strength_of_beam(V_n):
444     """
445     Design shear strength
446     Args:
447     V_n -Nominal shear strength of a cross-section
448     Return
449     V_d - Design shear strength in N
450     Note:
451     Reference:
452     IS 800:2007, cl. 8.4
453
454     """
455
456     gamma_m0 = IS800_2007.cl_5_4_1_Table_5["gamma_m0"]['yielding']
457     V_d = V_n / gamma_m0
458     return V_d
459
460
461 @staticmethod
462 def cl_8_4_1_nominal_plastic_shear_resistance_under_pure_shear(A_v,
463 f_yw):
464     """
465     Calculation of nominal plastic shear resistance under pure
466     shear
467
468     Args:
469     A_v - Shear area
470     f_yw - yield shear of the web
471     Returns:
472     V_n - Nominal plastic shear resistance under pure shear
473     Note:
474     Reference:
475     IS 800:2007, cl. 8.4.1
476
477     """
478
479     V_n = A_v * f_yw / math.sqrt(3)
480     return V_n
481
482 def cl_8_4_1_1_shear_area_of_different_section(A, b, d, h, t_f, t_w,
483 section, Axis_of_Bending, load_application_axis,
484 cross_section):
485     """
486     Calculation of shear area of different section
487
488     Args:
489     A - cross section area in mm**2
490     b - overall breadth of tubular section, breadth of I -
491     section flange in mm
492     d - clear depth of the web between flange in mm
493     h- overall depth of the section in mm
494     t_f - thickness of the flange in mm
495     t_w - thickness of the web in mm
496     section - Either 'I section ' and 'Channel Section' or '

```

```

5495 Rectangular hollow section of uniform depth'
5496         or 'Circular hollow tubes of uniform thickness' or
'plates' or 'solid bars'
5497         Load_application_axis - Either 'Loaded parallel to depth'
or 'Loaded parallel to width(b)'
5498         Axis_of_Bending - Either 'Major Axis Bending' or 'Minor
Axis Bending'
5499         cross_section - Either 'Hot Rolled' or 'Welded'
5500
5501     Return:
5502         A_v - Shear area in mm*mm
5503
5504     Note:
5505         Reference:
5506         IS 800:2007, cl. 8.4.1.1
5507
5508     """
5509     if section == 'I section' or section == 'Channel Section':
5510         if Axis_of_Bending == 'Major Axis Bending':
5511             if cross_section == 'Hot-Rolled':
5512                 A_v = h * t_w
5513             else:
5514                 A_v = d * t_w
5515             return A_v
5516
5517     if Axis_of_Bending == 'Minor Axis Bending':
5518         if cross_section == 'Hot-Rolled' or cross_section == 'Welded':
5519             A_v = 2 * b * t_f
5520             return A_v
5521     if section == 'Rectangular hollow section of uniform depth':
5522         if load_application_axis == 'Loaded parallel to depth':
5523             A_v = A * h / (b + h)
5524         else:
5525             A_v = A * b / (b + h)
5526         return A_v
5527
5528     if section == 'Circular hollow tubes of uniform thickness':
5529         A_v = A
5530         return A_v
5531
5532     if section == 'plates' or section == 'solid':
5533         A_v = A
5534         return A_v
5535
5536
5537     # cl8.4.2 TODO: CHECK RESISTANCE TO SHEAR BUCKLING
5538     # cl8.4.2.1 Check for resistance to shear buckling
5539     @staticmethod
5540     def cl_8_4_2_shear_buckling_check(d, t_w, k_v, fy, stiffeners):
5541         """
5542         Check for resistance against shear buckling
5543         Args:
5544             d - clear depth of web between flanges
5545             t_w - thickness of web
5546             k_v -shear buckling coefficient
5547             fy - yield stress in N/mm^2
5548             stiffeners - True if web has stiffeners else False
5549         Return:
5550             check - True if check is satisfied else false
5551         Note:

```



```

552         Reference:
553         IS 800:2007, cl. 8.4.2.1
554     """
555     epsilon = math.sqrt(250/fy)
556     if not stiffeners:
557         val = 67 * epsilon
558
559     else:
560         val = 67 * epsilon * math.sqrt(k_v/5.35)
561     if d / t_w > val :
562         check = True
563     else:
564         check = False
565
566     return check
567
568 # cl8.4.2.2 Shear buckling design method
569 @staticmethod
570 def cl_8_4_shear_buckling_coeff_Kv(only_at_support, c=None, d=None):
571     """
572     Args:
573         only_at_support - True if transverse stiffeners are provided
574                        only at support
575                        else False
576         c - spacing of transverse stiffeners
577         d - depth of web
578     Returns:
579         k_v - shear buckling coefficient
580     Note:
581         Reference - IS800_2007 cl.8.4.2.1 and cl.8.4.2.2
582     """
583     if only_at_support:
584         k_v = 5.35
585     elif c / d < 1:
586         k_v = 4 + 5.35 / (c / d) ** 2
587     else:
588         k_v = 5.35 + 4 / (c / d) ** 2
589     return k_v
590
591 @staticmethod
592 def cl_8_4_2_2_nominal_shear_post_critical(A_v,k_v,mu,E,d,t_w,f_yw):
593     """
594     Calculates nominal shear strength as governed by buckling using
595     simple post critical method
596     Args:
597         A_v - shear area defined in cl 8.4.1.1
598         k_v - shear buckling coefficient
599         mu - poisson's ratio
600         E - modulus of elasticity
601         d - depth of web
602         t_w - thickness of web
603         f_yw - characteristic yield stress of web material
604     Return:
605         V_n - nominal shear strength
606     Note:
607         Reference: IS800:2007 cl.8.4.2.2
608     """
609     T_cr_c = (k_v * math.pi ** 2 * E) / (12 * (1 - mu ** 2) * (d / t_w)

```

```

610     if lambda_w < 0.8:
611         T_b = (f_yw / math.sqrt(3))
612     elif 0.8 < lambda_w < 1.2:
613         T_b = (1 - 0.8 * max((lambda_w - 0.8), 0)) * (f_yw / math.sqrt
(3))
614     else:
615         T_b = f_yw / (math.sqrt(3) * lambda_w ** 2)
616
617     V_cr = A_v * T_b
618     V_n = V_cr
619     return V_n
620
621     @staticmethod
622     def cl_8_4_2_2_nominal_shear_tension_field(A_v, d, t_w, t_f, b_f, f_yw, c,
f_yf, N_f):
623         """
624         Calculates nominal shear strength as governed by buckling using
tension field method
625         Args:
626             A_v - shear area defined in cl 8.4.1.1
627             k_v - shear buckling coefficient
628             mu - poisson's ratio
629             E - modulus of elasticity
630             d - depth of web
631             t_w - thickness of web
632             t_f - thickness of flange
633             b_f - width of flange
634             f_yw - characteristic yield stress of web material
635             c - spacing of stiffeners in web
636             f_yf - characteristic yield stress of flange material
637             N_f - axial force in flange due to overall bending and
external axial force
638         Return:
639             Note: Reference - IS800:2007 cl.8.4.2.2
640         """
641         if c/d < 1.0:
642             return 'error : c/d must be greater than or equal to 1'
643
644         phi = math.atan(d / c) / 1.5
645         psi = 1.5 * T_b * math.sin(2 * phi)
646
647         gamma_m0 = IS800_2007.cl_5_4_1_Table_5["gamma_m0"]['yielding']
648         M_fr = 0.25 * b_f * t_f ** 2 * f_yf * (1 - (N_f / (b_f * t_f * f_yf
/ gamma_m0)) ** 2)
649
650         s = min(c, (2 / math.sin(phi) * math.sqrt(M_fr / f_yw * t_w)))
651         s_c = s
652         s_t = s
653         w_tf = d * math.cos(phi) - (c - s_c - s_t) * math.sin(phi)
654         f_v = math.sqrt(f_yw ** 2 - 3 * T_b ** 2 + psi ** 2) - psi
655         V_p = A_v * f_yw / math.sqrt(3)
656         V_tf = min(V_p, (A_v * T_b + 0.9 * w_tf * t_w * f_v * math.sin(phi)
))
657
658         V_tf = V_n
659         return V_n
660
661
662     # Stiffened web Panels
663     # End plate Design
664     # cl8.5.3 Anchor forces

```

```

665 @staticmethod
666 def cl_8_5_3_anchor_forces(d, t, f_y, V, V_cr, V_tf):
667     """ Calculation of resultant longitudinal shear and moment
668     Args:
669         d - web depth in mm
670         t - thickness of the section in mm
671         f_y - yield stress in N/mm**2
672         V_cr - critical shear strength as defined in cl8.4.2.2.
673         V_tf - basic shear strength as defined in cl8.4.2.2.
674
675         V - actual factored shear force
676     Return:
677         M_tf - resultant longitudinal moment in N*mm
678         R_tf - resultant longitudinal shear in N*mm
679
680     Note:
681         Reference:
682         IS 800:2007, cl. 8.5.3
683     """
684
685     V_p = d * t * f_y / math.sqrt(3)
686
687     H_q = 1.25 * V_p * math.sqrt(1 - V_cr / V_p)
688
689     if V < V_tf:
690         H_q *= (V - V_cr) / (V_tf - V_cr)
691
692
693     R_tf = H_q / 2
694     M_tf = H_q * d / 10
695
696     return (R_tf, M_tf)
697
698 # cl8.6 Design of Beams and Plate Girders with Solid Webs
699 # cl8.6.1 Minimum Web Thickness
700 # cl8.6.1.1 Serviceability requirement
701 @staticmethod
702 def cl_8_6_1_1_minimum_web_thickness(d, t_w, c, f_yw,
serviceability_requirement, web_connection_to_flange):
703     """
704     Checking the serviceability requirement of minimum thickness
of web
705     Args:
706         d - web depth in mm
707         t_w - thickness of web in mm
708         c - spacing of transverse stiffener in mm
709         epsilon_w - yield stress ratio of web
710         f_yw - yield stress of the web in N/mm**2
711         serviceability_requirement - '
transverse_stiffener_not_provided',
712                                     ,
only_transverse_stiffeners_provided_in_web_flange_connection_along_both_longitudinal_ed
,
713                                     ,
transverse_and_longitudinal_stiffener_at_one_level_as_cl_8_7_13'
714                                     ,
second_longitudinal_stiffener_provided_at_NA'
715         web_connection_to_flange - 'along_both_longitudinal_edges'
716                                     'along_one_longitudinal_edge'
717 >
718     Return:

```

```

719         True, if safety condition is satisfied else False
720     Note:
721         Reference:
722         IS 800:2007, cl. 8.6.1.1
723
724     """
725     epsilon_w = math.sqrt(250 / f_yw)
726
727     if serviceability_requirement == 'transverse_stiffener_not_provided
':
728         if web_connection_to_flange == 'along_both_longitudinal_edges':
729             return d/t_w <= 200 * epsilon_w
730         elif web_connection_to_flange == 'along_one_longitudinal_edge':
731             return d/t_w <= 90 * epsilon_w
732         elif serviceability_requirement == '
only_transverse_stiffeners_provided_in_web_flange_connection_along_both_longitudinal_ed
':
733             if 3 * d >= c >= d:
734                 return d / t_w <= 200 * epsilon_w
735             elif 0.74 * d <= c < d:
736                 return c / t_w <= 200 * epsilon_w
737             elif c < 0.74 * d:
738                 return d / t_w <= 270 * epsilon_w
739             else:
740                 return 'web_is_unstiffened'
741
742         elif serviceability_requirement == '
transverse_and_longitudinal_stiffener_at_one_level_as_cl_8_7_13':
743             if d < c <= 2.4 * d:
744                 return bool(d / t_w <= 250 * epsilon_w)
745             elif 0.74 * d <= c < d:
746                 return bool(c / t_w <= 250 * epsilon_w)
747             else:
748                 return bool(d / t_w <= 340 * epsilon_w)
749
750     else:
751         return bool( d / t_w <= 400 * epsilon_w)
752
753     # cl 8.6.1.2 Compression flange buckling requirement
754     def cl_8_6_1_2_web_thickness_to_avoid_buckling_of_compression_flange(d,
t_w, c, f_yf, Transverse_stiffener):
755         """
756         Check for minimum web thickness to avoid buckling of
compression flange
757         Args:
758             d - depth of the web in mm
759             t_w - thickness of the web in mm
760             c - spacing of transverse stiffener in mm
761             epsilon_f - yield stress ratio of flange
762             f_yw - yield stress of compression flange in N/mm**2
763             Transverse_stiffener - either 'provided' or 'not provided'
764         Return:
765             True or False
766         Note:
767             Reference:
768             IS 800:2007, cl. 8.6.1.1
769
770         """
771         epsilon_f = math.sqrt(250 / f_yf)
772         if Transverse_stiffener == 'not provided':
773             return bool( d / t_w <= 345 * epsilon_f ** 2)

```

```

774     else:
775         if c >= 1.5 * d:
776             return bool(d / t_w <= 345 * epsilon_f ** 2)
777         else:
778             return bool(d / t_w <= 345 * epsilon_f)
779
780     # cl8.7.1.5 Buckling resistance of stiffeners
781     # Effective length for load carrying web stiffeners
782     @staticmethod
783     def cl_8_7_1_5_effective_length_for_load_carrying_web_stiffeners(L,
784     restrained_condition):
785         """
786         Calculation of Effective length for load carrying web
787         stiffeners for calculating
788         buckling resistance F_xd
789         Args:
790             L - length of stiffener in mm
791             restrained_condition - Either '
792             flange_restrained_against_rotation' or
793             'flange_not_restrained_against_rotation
794             ,
795         Returns:
796             K_L - effective length for load carrying web stiffeners in
797             mm
798         Note:
799             Reference:
800             IS 800:2007, cl 8.7.1.5
801         """
802         if restrained:
803             K_L = 0.7 * L
804             return K_L
805         else:
806             K_L = L
807             return K_L
808
809     # Cl 8.7.2.4 Minimum stiffeners
810     @staticmethod
811     def
812     cl_8_7_2_4_I_s_for_transverse_web_Stiffeners_not_subjected_to_external_load
813     (c, d, t_w):
814         """
815         Calculation of second moment of area when transverse web
816         stiffener
817         not subjected to external load
818         Args:
819             d - depth of thw web in mm
820             t_w - minimum required web thickness foe spacing using
821             tension filed action ,as given in cl8.4.2.1 in mm
822             c - actual stiffener spacing in mm
823         Return:
824             I_s_min - second moment of area in mm**4
825         Note:
826             Reference:
827             IS 800:2007, cl 8.7.2.4
828         """
829         if c / d >= math.sqrt(2):
830             I_s_min = 0.75 * d * t_w ** 3
831         else:
832             I_s_min = 1.5 * d ** 2 * t_w ** 3 / c ** 2

```

```

826
827     return I_s_min
828
829 # cl 8.7.2.5 Buckling check on intermediate transverse web stiffeners
830 def cl_8_7_2_5_buckling_check_on_intermediate_transverse_web_stiffener(
V, F_qd, F_x, F_xd, M_q, M_yq, V_cr):
831     """
832         Buckling check on intermediate transverse web stiffeners
833     Args:
834         F_qd - design resistance of the intermediate stiffeners in
N
835         V -factored shear force adjacent to the stiffener in N
836         F_qd - deign resistance of an intermediate web stiffener
837             to buckling corresponding to buckling about at
838             axis parallel to the web as in cl 8.7.1.5 in N
839         F_x - external load or reaction at the stiffener in N
840         F_xd - design resistance of a load carrying stiffener
841             corresponding to buckling about axis parallel
842             to the web as in cl 8.7.1.5 in N
843         M_q - moment on the stiffener due to eccentrically
844             applied load anf transverse load, if any in N*mm
845         M_yq - yield moment capacity og the stiffener based
846             on its elastic modulus about its centroidal
847             axis parallel to the web in N*mm
848     Return:
849         F_q - stiffener force in N
850
851     Note:
852         Reference:
853         IS 800:2007, cl 8.7.2.5, cl.8.7.3
854
855     """
856     gamma_m0 = IS800_2007.cl_5_4_1_Table_5["gamma_m0"]['yielding']
857     F_q = min(V - V_cr / gamma_m0, F_qd)
858     if (F_q - F_x) / F_qd + F_x / F_xd + M_q / M_yq <= 1:
859         return F_q
860     else:
861         return 'cl.8.7.2.5.warning:buckling check on intermediate
transverse web stiffener not satisfied'
862
863
864 # cl 8.7.2.6 Connection of intermediate stiffeners to web
865 @staticmethod
866 def cl_8_7_2_6_shear_between_each_component_of_stiffener_and_web(t_w,
b_s, s_e):
867
868     """
869         Calculation of minimum allowable shear between each component
of stiffener and web
870     Args:
871         t_w - web thickness in mm
872         b_s - outstanding width of the stiffeners in mm
873
874     Returns:
875         V_is_min - minimum shear between each component of
stiffener and web in N/mm
876
877
878     Notes:
879         Reference:
880         IS 800:2007, cl 8.7.2.6

```

```

881
882     """
883     V = t_w ** 2 / 5 * b_s + s_e
884     return V
885
886
887 # cl 8.7.3 Load Carrying stiffeners
888 # cl 8.7.3.1 Web Checking
889 @staticmethod
890 def cl_8_7_3_1_area_of_cross_section_of__web(b_1, n_1, t_w):
891     """
892     Calculation of area of cross section of the web
893     Args:
894         b_1 - width of stiff bearing on the flange in mm
895         n_1 - dispersion of the load through the web
896             45 degree, to the level of half the depth
897             of the cross section
898         t_w - web thickness in mm
899             45 degrees, to the level of half the depth
900             of the cross section
901     Returns:
902         A_w - area of cross section of the web in mm**2
903     Notes:
904         Reference:
905         IS 800:2007, cl 8.7.3.1
906     """
907     A_w = (b_1 + n_1) / t_w
908     return A_w
909
910 # cl 8.7.4 Bearing Stiffeners
911
912 @staticmethod
913 def cl_8_7_4_force_applied_through_flange_by_loads_(b_1, n_2, t_w, f_yw
914 ):
915     """
916     Calculation of force applied through a flange by load or
917     reaction
918     exceeding the local capacity of the web at its connection
919
920     Args:
921         b_1 - stiff bearing length in mm
922         n_2 - length obtained by dispersion through the flange to
923             the web
924             junction at a slope of 1:2.5 to the plane of the
925             flange in mm
926         t_w - thickness of the web in mm
927         f_yw - yield stress of the web in N/mm**2
928
929     Returns:
930         F_w - force applied through flange by loads in N
931
932     Notes:
933         Reference:
934         IS 800:2007, cl 8.7.4
935     """
936     gamma_m0 = IS800_2007.cl_5_4_1_Table_5["gamma_m0"]['yielding']
937     F_w = (b_1 + n_2) * t_w * f_yw / gamma_m0
938     return F_w
939
940 # cl 8.7.5 Design of load carrying stiffeners

```

```

938 # cl 8.7.5.1 Buckling check
939 # cl 8.7.5.2 Bearing check
940
941 @staticmethod
942 def cl_8_7_5_2_bearing_strength_of_stiffeners(F_x, A_q, f_yq):
943     """
944         Calculation of bearing strength of stiffeners
945
946         Args:
947             F_x - external load or reaction in N
948             A_q - area of the stiffeners in contact with the flange in
949             mm
950             f_yq - yield stress of the stiffeners in N/mm**2
951
952         Return:
953             F_psd - bearing strength of stiffeners in N
954
955         Notes:
956             Reference:
957             IS 800:2007, cl 8.7.4
958     """
959     gamma_m0 = IS800_2007.cl_5_4_1_Table_5["gamma_m0"]['yielding']
960     F_psd = max(A_q * f_yq / (0.8 * gamma_m0), F_x)
961     return F_psd
962
963 # cl 8.7.9 Torsional Stiffeners
964 @staticmethod
965 def cl_8_7_9_minimum_second_moment_of_area_of_the_stiffener_section(D,
966 T_cf, L_LT, r_y):
967     """
968         calculation of minimum second moment of area of the stiffener
969
970         Args:
971             D = overall depth of beam at support in mm
972             T_cf = maximum thickness of compression flange in the span
973             under consideration in mm
974             K_L= laterally unsupported effective length of the
975             compression flange of the beam in mm
976             r_y = radius of gyration of the beam about the minor axis
977             in mm
978
979         Returns:
980             I_s_min = calculation of minimum second moment of area of
981             the stiffener in mm**4
982
983         Notes:
984             Reference:
985             IS 800:2007, cl 8.7.4
986     """
987     if L_LT / r_y <= 50:
988         alpha_s = 0.006
989     elif 50 < L_LT / r_y <= 100:
990         alpha_s = 0.3 / (L_LT / r_y)
991     else:
992         alpha_s = 30 / (L_LT / r_y) ** 2
993
994     I_s_min = 0.34 * alpha_s * D ** 3 * T_cf
995
996     return I_s_min

```


Appendix D

Python Code for Annex E IS: 800 2007

```
1 """ ANNEX E ELASTIC LATERAL TORSIONAL BUCKLING """
2 import math
3 # Table 5 Partial Safety Factors for Materials, gamma_m (dict)
4 class IS800_2007(object):
5     cl_5_4_1_Table_5 = {"gamma_m0": {'yielding': 1.10, 'buckling': 1.10},
6                          "gamma_m1": {'ultimate_stress': 1.25},
7                          "gamma_mf": {'shop': 1.25, 'field': 1.25},
8                          "gamma_mb": {'shop': 1.25, 'field': 1.25},
9                          "gamma_mr": {'shop': 1.25, 'field': 1.25},
10                         "gamma_mw": {'shop': 1.25, 'field': 1.50}
11                         }
12     # CALCULATION OF EFFECTIVE LENGTH AGAINST LATERAL TORSIONAL BUCKLING.
13     def cl_8_3_1_Effective_length_for_simply_supported_beams(L, D,
14                     Restraint_Condition_1, Restraint_Condition_2,
15                     Loading_Condition):
16         """
17         Calculate effective length against lateral torsional buckling
18         for simply supported Beams and girders
19         where no lateral restraint to the compression flange is
20         provided as per cl.8.3.1
21
22         Args:
23         L - Span of simply supported beams and girders in mm (
24         float)
25         D - Overall depth of the beam in mm (float)
26
27         Restraint_Condition - Either "Torsional Restraint" or "
28         wrapping Restraint"
29         Restraint_Condition_1 - "Torsional Restraint"
30         Restraint_Condition_2 - "Warping Restraint"
31
32         "Torsional Restraint" - Either "Fully restrained" or
33         "
34         Partially_restrained_by_bottom_flange_support_condition" or
35         "
36         Partially_restrained_by_bottom_flange_support_condition"
37
38         "Warping Restraint" - Either "Both flange fully restrained"
39         or
40         "compression flange fully restrained"
```

```

34 or
35 "
36 Compression_flange_partially_restrained" or
37 "Warping_not_restrained_in_both_flange
38 "
39 Loading_Condition - Either "Normal" or " Destabilizing"
40
41
42 Returns:
43 L_LT - c1_8_3_1_Effective length for simply supported
44 Beams in mm (float)
45
46 Note:
47 References:
48 IS800:2007, Table 15 (c1 8.3.1)
49
50 ""
51
52 if Restraint_Condition_1 == "Fully Restrained":
53     if Restraint_Condition_2 == "Both flanges fully restrained":
54         if Loading_Condition == "Normal":
55             return 0.70 * L
56         else:
57             return 0.85 * L
58     if Restraint_Condition_2 == "Compression flange fully
59 Restrained":
60         if Loading_Condition == "Normal":
61             return 0.75 * L
62         else:
63             return 0.90 * L
64     if Restraint_Condition_2 == "Both flanges partially restrained"
65 :
66         if Loading_Condition == "Normal":
67             return 0.80 * L
68         else:
69             return 0.95 * L
70     if Restraint_Condition_2 == "Compression flange partially
71 Restrained":
72         if Loading_Condition == "Normal":
73             return 0.85 * L
74         else:
75             return 1.00 * L
76     if Restraint_Condition_2 == "Wrapping not restrained in both
77 flanges":
78         if Loading_Condition == "Normal":
79             return 1.00 * L
80         else:
81             return 1.20 * L
82     if Restraint_Condition_1 == "Partially restrained by bottom flange
83 support connection":
84         if Restraint_Condition_2 == "Wrapping not restrained in both
85 flages":
86             if Loading_Condition == "Normal":
87                 return 1.00 * L + 2 * D
88             else:
89                 return 1.20 * L + 2 * D
90     if Restraint_Condition_1 == "Partially restrained by bottom flage

```

```

bearing support":
85     if Restraint_Condition_2 == "Wrapping not restrained in both
flages":
86         if Loading_Condition == "Normal":
87             return 1.2 * L + 2 * D
88         else:
89             return 1.4 * L + 2 * D
90
91     # Effective length for cantilever Beam
92
93     def cl_8_3_3_Table_16_Efective_length_for_cantiliver_beam(L, D,
94     Restraint_Condition_1, Restraint_Condition_2,
95     Loading_condition):
96         """
97         Calculate effective length for catiliver beam of projecting
98         length L as per cl.8.3.3
99
100         Args:
101             L - Projecting Length of cantiliver beam in mm (float)
102             D - Overall depth of he beam in mm (float)
103
104             Restrained_condition - Either "At support" or "At Top"
105
106             Restraint_Condition_1- "At support"
107             Restraint_Condition_2- "At Top"
108
109             At_support - Either "continous, with lateral restraint to
110             top"
111                         or "continous, with partial torsional
112             restraint"
113                         or "continous, with lateral and tosional
114             restraint "
115                         or "Restrained laterally,torsionally and
116             against rotation on plan "
117
118             At_top - Either "free"
119                     or "lateral restraint to top flange"
120                     or "Torsional restraint"
121                     or "Lateral and torsional restraint"
122
123             Loading_condition - Either "Normal" or "Destablizing"
124
125         Returns:
126             L_LT =
127         cl_8_3_3_Table_16_Efective_length_for_cantiliver_beam
128         Note:
129             References:
130             IS800:2007, Table 16 (cl 8.3.3)
131         """
132
133         if Restraint_Condition_1 == "Continuous, with lateral restraint to
134         top flage":
135             if Restraint_Condition_2 == "Free":
136                 if Loading_condition == "Normal":
137                     return (3.0 * L + 0 * D)
138                 else:
139                     return (7.5 * L + 0 * D)
140             if Restraint_Condition_2 == "Lateral restraint to top flage":
141                 if Loading_condition == "Normal":
142                     return (2.7 * L + 0 * D)

```

```

135         else:
136             return (7.5 * L + 0 * D)
137     if Restraint_Condition_2 == "Torsional restraint":
138         if Loading_condition == "Normal":
139             return (2.4 * L + 0 * D)
140         else:
141             return (4.5 * L + 0 * D)
142     if Restraint_Condition_2 == "Lateral and Torsional restraint":
143         if Loading_condition == "Normal":
144             return (2.1 * L + 0 * D)
145         else:
146             return (3.6 * L + 0 * D)
147     if Restraint_Condition_1 == "Continuous,with partial torsional
restraint":
148         if Restraint_Condition_2 == "Free":
149             if Loading_condition == "Normal":
150                 return (2.0 * L + 0 * D)
151             else:
152                 return (5.0 * L + 0 * D)
153         if Restraint_Condition_2 == "Lateral restraint to top flange":
154             if Loading_condition == "Normal":
155                 return (1.8 * L + 0 * D)
156             else:
157                 return (5.0 * L + 0 * D)
158         if Restraint_Condition_2 == "Torsional restraint":
159             if Loading_condition == "Normal":
160                 return (1.6 * L + 0 * D)
161             else:
162                 return (3.0 * L + 0 * D)
163         if Restraint_Condition_2 == " Lateral and Torsional
restraint":
164             if Loading_condition == "Normal":
165                 return (1.4 * L + 0 * D)
166             else:
167                 return (2.4 * L + 0 * D)
168     if Restraint_Condition_1 == "Continuous,with lateral and torsional
restraint":
169         if Restraint_Condition_2 == "Free":
170             if Loading_condition == "Normal":
171                 return (1.0 * L + 0 * D)
172             else:
173                 return (2.5 * L + 0 * D)
174         if Restraint_Condition_2 == "Lateral restraint to top flange":
175             if Loading_condition == "Normal":
176                 return (0.9 * L + 0 * D)
177             else:
178                 return (2.5 * L + 0 * D)
179         if Restraint_Condition_2 == "Torsional restraint":
180             if Loading_condition == "Normal":
181                 return (0.8 * L + 0 * D)
182             else:
183                 return (1.5 * L + 0 * D)
184         if Restraint_Condition_2 == " Lateral and Torsional restraint":
185             if Loading_condition == "Normal":
186                 return (0.7 * L + 0 * D)
187             else:
188                 return (1.2 * L + 0 * D)
189     if Restraint_Condition_1 == "Restrained laterally,torsionally and
against rotation on plan":
190         if Restraint_Condition_2 == "Free":
191             if Loading_condition == "Normal":

```

```

192         return (0.8 * L + 0 * D)
193     else:
194         return (1.4 * L + 0 * D)
195     if Restraint_Condition_2 == "Lateral restraint to top flage":
196         if Loading_condition == "Normal":
197             return (0.7 * L + 0 * D)
198         else:
199             return (1.4 * L + 0 * D)
200     if Restraint_Condition_2 == "Torsional restraint":
201         if Loading_condition == "Normal":
202             return (0.6 * L + 0 * D)
203         else:
204             return (0.6 * L + 0 * D)
205     if Restraint_Condition_2 == "Lateral and Torsional restraint":
206         if Loading_condition == "Normal":
207             return (0.5 * L + 0 * D)
208         else:
209             return (0.5 * L + 0 * D)
210
211     def cl_8_3_Effective_length_against_torsional_restraint(L, D, Beam_type
212     , Restraint_Condition_1,
213     Restraint_Condition_2, Loading_Condition):
214         """
215         Calculation of effective length for given type of beam type as
216         per cl.8.3
217
218         Args:
219         L- Span of simply supported beams and girders in mm (float)
220         for
221         "
222         Simply_supported_with_no_lateral_restrained_to_the_compression_flanges
223         ",
224         Projecting Length of cantiliver beam in mm (float) for
225         "Cantilever_beam",
226         Length of relevent segment between the lateral restraint
227         in mm (float) for
228         "Simply_supported_with_intermediate_lateral_restraints",
229         Centre-to-centre distance of the restraint member in mm (
230         float) for
231         "
232         Beam_provided_with_members_to_give_effective_lateral_restrain_to_compression_flange_at_
233         "
234         D - Overall depth of he beam in mm (float)
235
236         Beam_type - Either "
237         Simply_supported_with_no_lateral_restrained_to_the_compression_flanges"
238         or "
239         Simply_supported_with_intermediate_lateral_restraints"
240         or "
241         Beam_provided_with_members_to_give_effective_lateral_restrain_to_compression_flange_at_
242         "
243         or "Cantilever_beam"
244
245         FOR "
246         Simply_supported_with_no_lateral_restrained_to_the_compression_flanges"
247
248         Restraint_Condition - Either "Torsional Restraint" or "wrapping
249         Restraint"

```

```

237         Restraint_Condition_1- "Torsional Restraint"
238         Restraint_Condition_2- "Warping_Restraint"
239
240         "Torsional Restrained" - Either "Fully_resrtrained" or
241         "
242         Partially_restrained_by_bottom_flange_support_condition" or
243         "
244         Partially_restrained_by_bottom_flange_support_condition"
245         "Warping_Restraint" - Either "Both_flange_fully_restrained" or
246         "compression_flange_fully_restrained"
247         or
248         "
249         Compression_flange_partially_restrained" or
250         "Warping_not_restrained_in_both_flange
251         "
252
253         FOR "Cantilever_beam"
254
255         Restrained_condition - Either "At support" or "At Top" for "
256         Cantilever_beam"
257
258         Restraint_Condition_1- "At support"
259         Restraint_Condition_2- "At Top"
260
261         At_support - Either "continous, with lateral restraint to top"
262         or "continous, with partial torsional
263         restraint"
264         or "continous, with lateral and tosional
265         restraint "
266         or "Restrained laterally,torsionally and
267         against rotation on plan "
268
269         At_top - Either "free"
270         or "lateral restraint to top flange"
271         or "Torsional restraint"
272         or "Lateral and torsional restraint"
273
274         Loading_condition - Either "Normal" or "Destablizing"
275
276
277     Returns :
278         L_LT - Effective_length_of_beam in m (float)
279
280     Note:
281         References:
282         IS800:2007, cl 8.3.
283
284     """
285
286     if Beam_type == "
287     Simply_supported_with_no_lateral_restrained_to_the_compression_flanges"
288     :
289         L_LT = cl_8_3_1_Effective_length_for_simply_supported_beams(L,
290         D, Restraint_Condition_1,
291         Restraint_Condition_2, Loading_Condition)
292     elif Beam_type == "

```

```

Simply_supported_with_intermediate_lateral_restraints":
285     L_LT = 1.2 * L
286     elif Beam_type == "
Beam_provided_with_members_to_give_effective_lateral_restrain_to_compression_flange_at_
":
287         L_LT = 1.2 * L
288     else:
289         L_LT = cl_8_3_3_Table_16_Efective_length_for_cantilever_beam(L,
D, Restraint_Condition_1,
290 Restraint_Condition_2, Loading_Condition)
291
292     return L_LT
293
294     # E-1 ELASTIC CRITICAL MOMENT
295
296     # E-1.1 General
297     # TODO:Calculate L_LT = L_LT =
cl_8_3_Effective_length_against_lateral_torsional_buckling(L,D,
Beam_type,Restraint_Condition_1,Restraint_Condition_2,Loading_Condition
)
298
299     def
Annex_E_1_1_elastic_critical_moment_corresponding_to_lateral_torsional_buckling_of_doub
(
300         I_y, I_w, I_t, G, E, L_LT):
301
302         """
303             Calculate the elastic critical moment corresponding to lateral
304             torsional buckling of a doubly symmetric prismatic beam
305             subjected to uniform
306             moment in the unsupported length and torsionally
307             restraining lateral supports as per Annex E-1.1
308
309             Args:
310             I_y - Moment of inertia about the minor axis (in quartic mm
) (float)
311             I_w - Warping constant of the cross- section (mm**4)(float)
312             I_t - St. Venants torsion constant of the cross-section (mm
**4)(float)
313             G - Modulus of rigidity (float)
314             L_LT =
cl_8_3_Effective_length_against_lateral_torsional_buckling(L,D,
Beam_type,Restraint_Condition_1,Restraint_Condition_2,Loading_Condition
)
315
316             Returns:
317             M_cr - Elastic critical moment corresponding to lateral
torsional
318             buckling of doubly symmetric prismatic beam ( in N*
mm) (float)
319
320             Note:
321             Reference:
322             IS800:2007, Annex - E-1.1
323
324             """
325
326     sum_value = (I_w / I_y) + (G * I_t * L_LT * L_LT) / (pi * pi * E *

```



```

I_y)
327
328     M_cr = ((pi * pi * E * I_y) / (L_LT * L_LT)) * ((sum_value) ** 0.5)
329
330     return M_cr
331
332     def Annex_E_Table_42_constant_c_1_c_2_c_3(Loadings_and_Support_condition
, si, K):
333         """
334             Calculate Value of constant c_1,c_2,c_3 as per Table_42 Annex-E
335             Args:
336                 Loadings_and_Support_condition - Either "Simply supported
with ends moments (M,si*M)"
337                                                     or "Simply Supported beam
with UDL"
338                                                     or "Fixed support with UDL"
339                                                     or "Simply Supported beam
with point load at centre"
340                                                     or "Fixed Supported beam
with point load at centre"
341                                                     or "Simply Supported beam
with point at L/4 distance from both ends"
342
343             Bending_Moment_diagram- BMD for "Simply supported with ends
moments (M,si*M)" by varying value of 'si' as-
344                                                     si - [
+1,+3/4,+1/2,+1/4,0,-1/4,-1/2,-3/4,-1 ]
345                                                     BMD for "Simply Supported beam with
UDL"
346                                                     si - 0 (Assumed value)
347                                                     BMD for "Fixed support with UDL"
348                                                     si - 0 (Assumed value)
349                                                     BMD for "Simply Supported beam with
point load at centre"
350                                                     si - 0 (Assumed value)
351                                                     BMD for "Fixed Supported beam with
point load at centre"
352                                                     si - 0 (Assumed value)
353                                                     BMD for "Simply Supported beam with
point at L/4 distance from both ends"
354                                                     si - 0 (Assumed value)
355
356             K - [1.0,0.7,0.5]
357
358             Returns:
359                 Value of constant c_1,c_2,c_3
360
361             Note:
362                 References:
363                 IS800:2007,Table 42,c1 E-1.2
364             """
365
366             if Loadings_and_Support_condition == "Simply supported with ends
moments (M,si*M)":
367                 if si == 1:
368                     if K == 1.0:
369                         return {"c1": 1.000, "c2": 0, "c3": 1.000}
370                     if K == 0.7:
371                         return {"c1": 1.000, "c2": 0, "c3": 1.113}
372                     if K == 0.5:
373                         return {"c1": 1.000, "c2": 0, "c3": 1.144}

```

```

374     if si == +3 / 4:
375         if K == 1.0:
376             return {"c1": 1.141, "c2": 0, "c3": 0.998}
377         if K == 0.7:
378             return {"c1": 1.270, "c2": 0, "c3": 1.565}
379         if K == 0.5:
380             return {"c1": 1.305, "c2": 0, "c3": 2.283}
381     if si == +1 / 2:
382         if K == 1.0:
383             return {"c1": 1.323, "c2": 0, "c3": 0.992}
384         if K == 0.7:
385             return {"c1": 1.473, "c2": 0, "c3": 1.556}
386         if K == 0.5:
387             return {"c1": 1.514, "c2": 0, "c3": 2.271}
388     if si == +1 / 4:
389         if K == 1.0:
390             return {"c1": 1.879, "c2": 0, "c3": 0.939}
391         if K == 0.7:
392             return {"c1": 2.092, "c2": 0, "c3": 1.473}
393         if K == 0.5:
394             return {"c1": 2.150, "c2": 0, "c3": 2.150}
395     if si == 0:
396         if K == 1.0:
397             return {"c1": 1.563, "c2": 0, "c3": 0.977}
398         if K == 0.7:
399             return {"c1": 1.739, "c2": 0, "c3": 1.531}
400         if K == 0.5:
401             return {"c1": 1.788, "c2": 0, "c3": 2.235}
402     if si == -1 / 4:
403         if K == 1.0:
404             return {"c1": 2.281, "c2": 0, "c3": 0.855}
405         if K == 0.7:
406             return {"c1": 2.538, "c2": 0, "c3": 1.340}
407         if K == 0.5:
408             return {"c1": 2.609, "c2": 0, "c3": 1.957}
409     if si == -1 / 2:
410         if K == 1.0:
411             return {"c1": 2.704, "c2": 0, "c3": 0.676}
412         if K == 0.7:
413             return {"c1": 3.009, "c2": 0, "c3": 1.059}
414         if K == 0.5:
415             return {"c1": 3.093, "c2": 0, "c3": 1.546}
416     if si == -3 / 4:
417         if K == 1.0:
418             return {"c1": 2.927, "c2": 0, "c3": 0.366}
419         if K == 0.7:
420             return {"c1": 3.009, "c2": 0, "c3": 0.575}
421         if K == 0.5:
422             return {"c1": 3.093, "c2": 0, "c3": 0.837}
423     if si == -1:
424         if k == 1.0:
425             return {"c1": 2.752, "c2": 0, "c3": 0}
426         if k == 0.7:
427             return {"c1": 3.063, "c2": 0, "c3": 0}
428         if k == 0.5:
429             return {"c1": 3.149, "c2": 0, "c3": 0}
430     if Loading_and_Support_condition == "simply Supported beam with UDL
":
431         if si == 0:
432             if K == 1.0:
433                 return {"c1": 1.132, "c2": 0.459, "c3": 0.525}

```

```

434         if K == 0.5:
435             return {"c1": 0.972, "c2": 0.304, "c3": 0.980}
436     if Loading_and_Support_condition == "Fixed support with UDL":
437         if si == 0:
438             if K == 1.0:
439                 return {"c1": 1.285, "c2": 1.562, "c3": 0.753}
440             if K == 0.5:
441                 return {"c1": 0.712, "c2": 0.652, "c3": 1.070}
442     if Loading_and_Support_condition == "Simply Supported beam with
point load at centre":
443         if si == 0:
444             if K == 1.0:
445                 return {"c1": 1.365, "c2": 0.553, "c3": 1.780}
446             if K == 0.5:
447                 return {"c1": 1.070, "c2": 0.432, "c3": 3.050}
448     if Loading_and_Support_condition == "Fixed Supported beam with
point load at centre":
449         if si == 0:
450             if K == 1.0:
451                 return {"c1": 1.565, "c2": 1.257, "c3": 2.640}
452             if K == 0.5:
453                 return {"c1": 0.938, "c2": 0.715, "c3": 4.800}
454     if Loading_and_Support_condition == "Simply Supported beam with
point at L/4 distance from both ends":
455         if si == 0:
456             if K == 1.0:
457                 return {"c1": 1.046, "c2": 0.430, "c3": 1.120}
458             if K == 0.5:
459                 return {"c1": 1.010, "c2": 0.410, "c3": 1.390}
460
461     # E-1.2 Elastic Critical Moment of a Section Symmetrical About Minor
Axis
462     # TODO: Calculate c_1,c_2,c_3 = Annex_E_Table_42_constant_c_1_c_2_c_3(
Loading_and_Support_condition,si,K)
463     def Elastic_critical_moment_of_a_section_symmetrical_about_minor_axis(
L_LT, c_1, c_2, c_3, E, K, K_w, y_g, A_e, b,
464                                                                                                     t
, I_fc, I_ft, I_y, G, h, h_L, h_y, n,
465
I_section=True, Open_section=True,
466
Plain_flange=False):
467         """
468         Calculate the elastic critical moment for lateral torsional
buckling
for beam which is
469         symmetrical only about the minor axis, and bending about major
axis as per Annex E-1.2
470
471         Args:
472             c_1,c_2,c_3- Annex_E_Table_42_constant_c_1_c_2_c_3(
Loading_and_Support_condition,si,K)
473             K - Effective length factors of the unsupported length
accounting for boundry condition
474                 at the end letral supports. It is analogous to the
effective length factirs for
475                 compression members with end rotational restraint.
476             K_w -Warping restraint factor.
477             y_g -y distance between the point of application of the
load and the shear centre of
478                 the cross-section and is positive when the load is
acting towards the shear

```

```

479         centre from the point of application.
480         y_s- co-ordinate of the shear centre with respect to
centriod,positive when the shear
481         centre is on the compression side of the centriod.
482         y,z- co-ordinate of the elemental area with respect to
centriod of the section
483         E - Youngs modulus of elasticity ( N per sq.mm)(float)
484         G- Modulus of regidity (float)
485         I_y - Moment of inertia about minor axis (in mm**4)(float)
486         I_fc - Moment of inertia of the compression flange about
minor axis of the entire section (in mm**4)(float)
487         I_ft - Moment of inertia of the tension flange about minor
axis of the entire section (in mm**4)(float)
488         I_w - The wraping constant either for "
I_section_mono_symmetric_about_weak_axis" or for
489
"Angle,Tee,
narrow_rectangle_section and approximetly for hollow_section"
490         I_t - Torsion constant either "for open_section" or "
hollow_section"
491         A_e - Area encloed by the section (in sq mm)(float)
492         b - Breadth of the elements of the section(mm)(float)
493         t - Thickness of the elements of the section (mm)(float)
494         h_L- Height of the lip in mm(float)
495         h- overall height of the section in mm(float)
496         h_y - Distance between shear centre of the two flange of
the cross-section in mm (float)
497         Flange type - Either "Plain_flange" or "Lipped_flange"
498         L_LT =
cl_8_3_Effective_length_against_lateral_torsional_buckling(L,D,
Beam_type,Restraint_Condition_1,Restraint_Condition_2,Loading_Condition
)
499         Returns:
500         M_cr-
Elastic_critical_moment_of_a_section_symmetrical_about_minor_axis (in N
*mm)(float)
501
502         Note:
503         References:
504         IS800:2007,c1 E-1.2
505         """
506
507         beta_f = I_fc / (I_ft + I_fc)
508
509         if I_section is True:
510             I_w = (1 - beta_f) * beta_f * I_y * h_y * h_y
511         else:
512             I_w = 0
513
514         if Plain_flange is True:
515             if beta_f > 0.5:
516                 y_j = 0.8 * (2 * beta_f - 1) * h_y / 2.0
517             else:
518                 y_j = 1.0 * (2 * beta_f - 1) * h_y / 2.0
519         else:
520             if beta_f > 0.5:
521                 y_j = 0.8 * (2 * beta_f - 1) * (1 + h_L / h_) * h_y / 2.0
522             else:
523                 y_j = (2 * beta_f - 1) * (1 + h_L / h) * h_y / 2
524
525         if Open_section is True:
526             sum_value = 0

```

```

527         for i in range(n - 1):
528             sum_value += (b * t * t * t) / 3
529             I_t = sum_value
530     else:
531         sum_value = 0
532         for i in range(n - 1):
533             sum_value += (b / t)
534
535     I_t = 4 * A_e / sum_value
536
537     T_1 = c_1 * (pi * pi * E * I_y) / (L_LT)
538     T_2 = (K / K_w) ** 2 * (I_w / I_y)
539     T_3 = (G * I_t * L_LT * L_LT) / (pi * pi * E * I_y)
540     T_4 = ((c_2 * y_g) - (c_3 * y_j)) ** 2
541     T_5 = ((c_2 * y_g) - (c_3 * y_j))
542
543     M_cr = T_1 * (((T_2 + T_3 + T_4) ** 0.5) - T_5)
544
545     return M_cr
546 # =====

```

Appendix E

Creating class for sectional properties of I section

```
1 import math
2
3 class I_sectional_Properties(object):
4     def __init__(self,D,B,t_w,t_f,alpha=90,r_1=0,r_2=0):
5         """ Calculation of sectional properties of built-up I section
6         Args:
7             D- depth of beam in mm
8             B- Width of flange in mm
9             t_w- thickness of web in mm
10            t_f- thickness of flange in mm
11            alpha- flange slope in degree
12            r_1-root radius in mm
13            r_2-toe radius in mm
14        Return:
15            A- Sectional Area of the section in mm**2
16            M - Mass (kg/m)
17            I_zz - second moment of area in cm**4
18            I_yy - second moment of area in cm**4
19            r_z - radius of gyration in cm
20            r_y - radius of gyration in cm
21            Z_ez - elastic modulus in cm**3
22            Z_ey - elastic modulus in cm**3
23            Z_pz - plastic modulus in cm**3
24            Z_py - plastic modulus in cm**3
25
26            """
27
28            self.A = ((2*B*t_f) + ((D-2*t_f)*t_w))/100
29            self.M = 7850 * self.A / 10000
30            self.I_zz = ((D - 2*t_f)**3 * t_w /12 + (B*t_f**3)/6+(B/2*t_f*(D-
31            t_f)**2))/10000
32            self.I_yy = ((D-2*t_f)*t_w**3 /12 + B**3*t_f/6)/10000
33            self.r_z = math.sqrt(self.I_zz / self.A)
34            self.r_y = math.sqrt(self.I_yy / self.A)
35            self.Z_ez = (self.I_zz * 2*10) / (D)
36            self.Z_ey = (self.I_yy * 2*10) / (B)
37            self.y_p = (((D - 2*t_f)**2*t_w/8 + B*t_f*(D-t_f)/2) / ((D-t_f)/2*
38            t_w + B*t_f ))/10
39            self.Z_pz = (2 * (self.A / 2 * self.y_p))
40            self.z_p = (((D-2*t_f)*t_w**2)/8 + (B*t_f*B)/4)/((D-2*t_f)*t_w/2 +
41            (B*t_f))
```

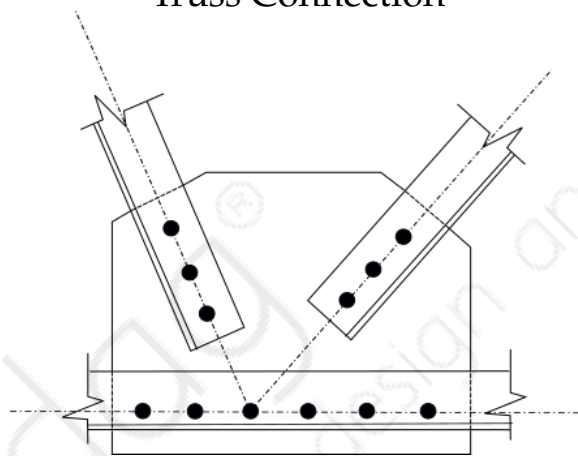
```
self.Z_py = 2 * (self.A / 2 * self.z_p)
```

Appendix F

DDCL for Bolted Truss Connection

Design and Detailing Checklist (DDCL)
and
Design and Detailing Query (DDQ)

Truss Connection



Prepared by:

Rachna Gupta

Under the guidance of:

Prof. Siddhartha Ghosh



Indian Institute of Technology, Bombay

July 10, 2019

Contents

1 Bolt Design	3
1.1 Gusset Plate	3
1.1.1 Gusset plate thickness (t_g)	3
1.2 Bolt Value	3
1.3 Bearing type bolts	3
1.3.1 Design strength of bolt (V_{db})	3
1.3.2 Shear capacity (V_{dsb})	3
1.3.3 Bearing capacity (V_{dpb})	4
1.4 Friction grip type bolts	4
1.4.1 Slip resistance (V_{dsf})	4
1.5 Bolt Capacity	4
1.6 Check of long joint	4
1.7 Check of large grip length	5
1.8 Bolt capacity	5
2 Number of Bolts	6
2.1 Minimum number of bolts (n)	6
2.2 Number of bolts in zero force member (n_0)	6
3 Gusset Plate Checks	7
3.1 Whitmore effective width (B_{eff})	7
3.1.1 For one bolt line	7
3.1.2 For staggered bolts	8
3.2 Check for Tension	8
3.2.1 Check for Tension Yielding	8
3.2.2 Check for Tension Rapture	8
3.3 Whitmore equivalent column length (l_e)	9
3.4 Check for Buckling Failure	9
4 Check for Truss Member	11
4.1 Block failure check	11
5 Detailing	12
5.1 Pitch (p) and Gauge (g)	12
5.2 End (e) and Edge (e')	13
5.3 Gusset plate Dimensions	13
5.3.1 Height of Gusset plate (h_g)	13
5.3.2 Width of gusset plate (w_g)	13

User Inputs

- Connecting members
Truss Member Sections*
- Types of Connectivity*
Bolted
- Factored loads
Axial Force (kN)
- Material Property of section
 f_u (MPa)*
 f_y (MPa)*
- Fasteners Details
Bolt Diameter (mm)*
Bolt Type *
Property class *
- Detailing
Angle of inclination (degree)*
- Plate
Thickness (mm)*
 f_u (MPa)*
 f_y (MPa)*

Design and Detailing Checks



Osdag®

Open steel design and graphics

Check 1

Bolt Design

The truss members and gusset plate are connected by bolts.

1.1 Gusset Plate

1.1.1 Gusset plate thickness (t_g)

Thickness of the gusset plate provided is usually equal to or slightly higher than members that are connected to gusset plate.

Thickness of gusset plate is calculated on the basis of maximum force acting on the truss members and is given by the table. [Reference: Table 5.12 Design of steel structure by N.Subramanian]

Maximum design force in diagonal(KN)	upto 200	200-450	450-750	750-1650	1650-2250	2250-3000
Thickness of gusset plate(mm)	8	10	12	16	18	20

1.2 Bolt Value

1.3 Bearing type bolts

1.3.1 Design strength of bolt (V_{db})

[Reference: Cl. 10.3.2, IS 800:2007]

The design strength of bolt is taken as the smaller of the value as governed by shear, V_{dsb} (1.3.2) and bearing V_{dpb} (1.3.3).

$$V_{db} = \min(V_{dsb}, V_{dpb}) \quad (1.1)$$

1.3.2 Shear capacity (V_{dsb})

[Reference: Cl. 10.3.3, IS 800:2007]

Currently, conservatively assuming all the shear planes passes through the threads of the bolt. Shear capacity of bearing bolts,

$$V_{dsb} = \frac{f_u n_n A_{nb}}{\sqrt{3} \gamma_{mb}} \quad (1.2)$$

Where,

V_{dsb} = design strength of bolt, as governed by shear strength;

f_u = ultimate tensile strength of a bolt;
 n_n = number of shear planes with threads intercepting the shear plane;
 A_{nb} = net shear area of the bolt at threads, taken as the area corresponding to root diameter at the thread;
 γ_{mb} = partial safety factor for bolt

1.3.3 Bearing capacity (V_{dpb})

[Reference: Cl. 10.3.4, IS 800:2007]

$$V_{dpb} = \frac{2.5 k_b d t f_u}{\gamma_{mb}} \quad (1.3)$$

Where,

k_b is smaller of $\frac{e}{3d_0}$, $\frac{p}{3d_0} - 0.25$, $\frac{f_{ub}}{f_u}$, 1.0;

e, p = end and pitch distances of the bolt along line of action respectively;

d, d_0 = diameter of bolt and bolt hole respectively;

t = summation of the thicknesses of the connected plates experiencing bearing stress in the same direction;

γ_{mb} = partial safety factor.

1.4 Friction grip type bolts

1.4.1 Slip resistance (V_{dsf})

[Reference: Cl. 10.4.3, IS 800:2007]

$$V_{nsf} = \frac{F_o \mu_f n_e K_h}{\gamma_{mf}} \quad (1.4)$$

μ_f = coefficient of friction (slip factor);

n_e = number of effective interfaces offering frictional resistance to slip;

$K_h = 1.0$ for bolts in clearance holes and 0.85 for bolts in oversized holes;

F_o = proof load = $A_{nb} f_0$;

f_0 = proof stress $0.7 A_{ub}$;

γ_{mf} = partial safety factor.

1.5 Bolt Capacity

From here on, V_{bolt} is used where bolt capacity is considered. V_{bolt} is taken as V_{dsf} if friction grip bolt is considered and V_{db} , if bearing bolt is considered.

1.6 Check of long joint

[Reference: Cl. 10.3.3.1 IS 800 : 2007]

when $l_j \geq 15d$, design capacity will be reduced by factor β_{lj} .

$$\beta_{lj} = 1.075 - 0.005(l_j/d), 0.75 \leq \beta_{lj} \leq 1.0 \quad (1.5)$$

where,

l_j = length of joint (measured in the direction of load transfer);

d = nominal diameter of the bolt.

1.7 Check of large grip length

[Reference: Cl. 10.3.3.2 and Cl. 10.3.3.1 IS 800 : 2007]

when $l_g \geq 5d$, the design shear capacity is reduced by a factor β_{lg} .

$$\beta_{lg} = \frac{8}{3 + l_g/d} \quad (1.6)$$

where,

l_g = total thickness of the connected plates (grip length);

d = nominal diameter of the bolt.

Note: β_{lg} should not be greater than β_{lj}

1.8 Bolt capacity

$$V'_{db} = V_{bolt}\beta_{lj}\beta_{lg} \quad (1.7)$$

Check 2

Number of Bolts

2.1 Minimum number of bolts (n)

Number of bolts required for each truss member(n) is given by the equation;

$$n = \frac{P}{V_{bolt}} \quad (2.1)$$

If leg size of truss member section is than 130mm, then linear bolt raw will be provided. If leg size is greater than 130mm than either linear bolt raw or staggered bolt raw can be provided (chain bolting is usally not preferred as staggered bolting is more efficient).

2.2 Number of bolts in zero force member (n_0)

[Reference:CI 10.7 IS 800: 2007]

When no axial force is acting on the truss member then a force of at least 0.3times the member design capacity acts at the ends of tensile or compression member and minimum number of bolt is calculated on this basis.

$$F_d = \frac{f_y A}{\gamma_{m0}} \quad (2.2)$$

where,

F_d = Member design capacity;

$$n_0 = \frac{0.3F_d}{V_{bolt}} \quad (2.3)$$

Check 3

Gusset Plate Checks

3.1 Whitmore effective width (B_{eff})

[Reference: Page No. 365 Design of steel by structure N.Subramanian 13th Edition]

Maximum direct stress (compression or tension) in gusset plate from an individual member is estimated adequately by ensuring that the member force is distributed uniformly over an effective area given by 30° dispersion from outer row of fasteners as shown in figures:

3.1.1 For one bolt line

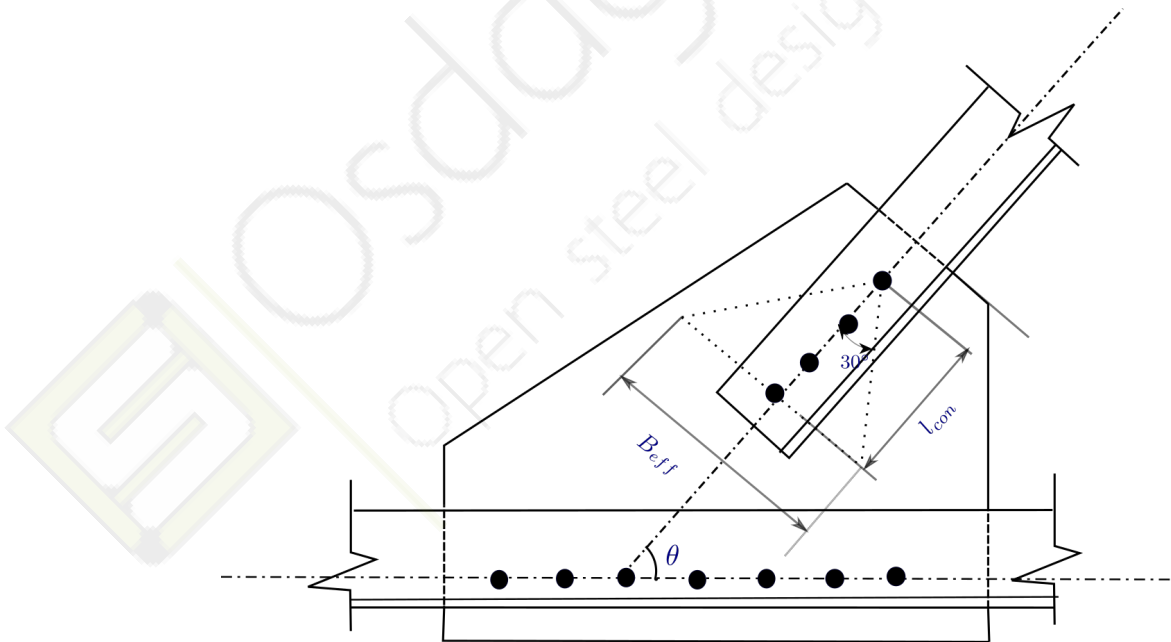


Figure 3.1: Whitmore Effective Width for Linear bolting arrangement

$$B_{eff} = 2l_{con}\tan(30^\circ) \quad (3.1)$$

3.1.2 For staggered bolts

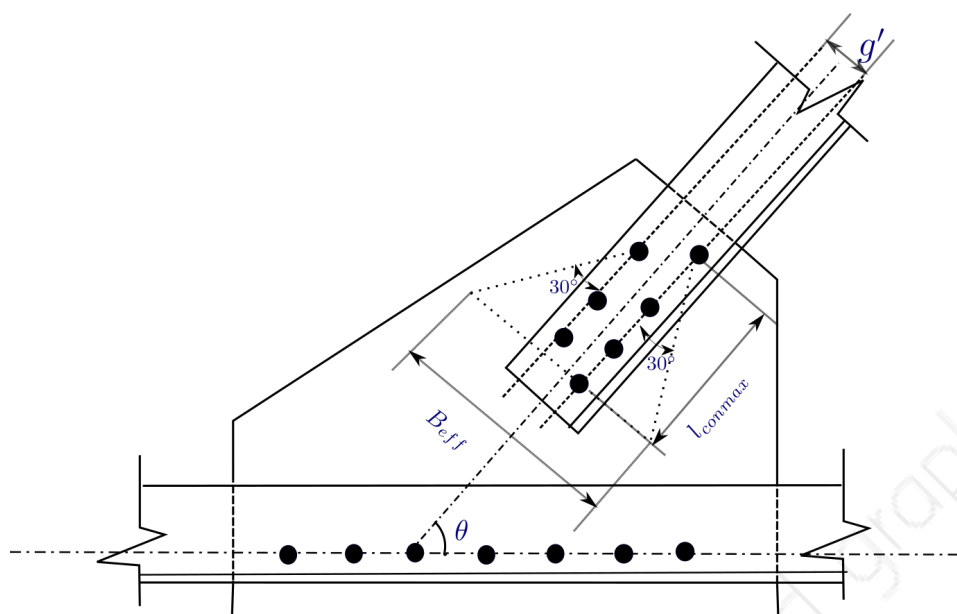


Figure 3.2: Whitmore Effective Width for Staggered bolting arrangement

$$B_{eff} = g' + (2l_{conmax}\tan(30^\circ)) \quad (3.2)$$

where,

g' = gauge length between the staggered bolts;

l_{conmax} = maximum connection length of staggered bolts for each member.

3.2 Check for Tension

3.2.1 Check for Tension Yielding

[Reference: Cl 6.2 IS 800: 2007]

$$\frac{P}{(B_{eff}t_g)} \leq \frac{f_y}{\gamma_{m0}} \quad (3.3)$$

where,

P = Factored Axial force on each member;

B_{eff} = Whitmore effective width;

t_g = Gusset plate thickness;

f_y = yield strength of plate;

γ_{m0} = Partial factor of safety for yielding.

3.2.2 Check for Tension Rapture

[Reference: Cl 6.3 IS 800: 2007]

$$\frac{P}{(B_{eff} - nd_0)t_g} \leq \frac{0.9f_u}{\gamma_{m1}} \quad (3.4)$$

where,

P = Axial force on each member;

d_0 = bolt hole diameter;

n = numbers of bolts

B_{eff} = Whitmore effective width;

t_g = Gusset plate thickness;

f_u = ultimate strength of plate;

γ_{m0} = Partial factor of safety.

3.3 Whitmore equivalent column length (l_c)

[Reference: Page No. 365 Design of steel by structure N.Subramanian 13th Edition]

Whitmore equivalent column length is defined as average of the three lengths projected from the whitmore section, section, in the direction of member, to the fastener lines of the adjoining members. The three lengths are taken at the two ends of the width and at the center. If Whitmore section intersects an adjoining member bolt line, the length at that end is assumed to be zero. Conservatively, Whitmore equivalent column length can also be taken as the distance from center of member to the fasteners lines of adjoining members.

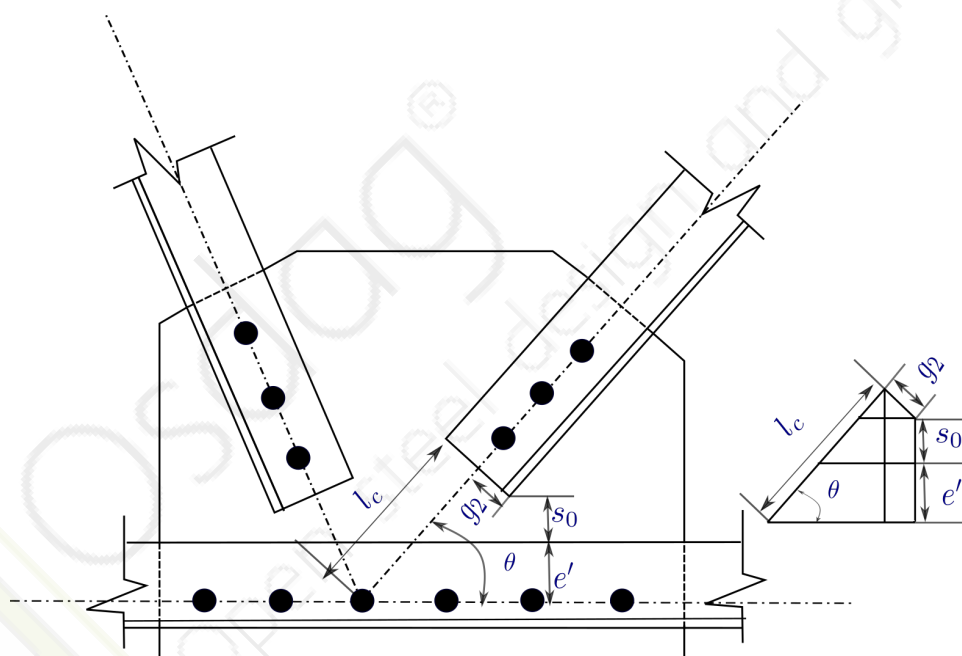


Figure 3.3: Equivalent Column Length

$$l_c = \frac{s_0 + g_1 + g_2 \cos(\theta)}{\sin(\theta)} \quad (3.5)$$

where,

s_0 = spacing between bottom chord and the connected truss member;

e' = edge distance of bottom chord;

g_2 = gauge length of connected truss member;

θ = angle of inclination of connected truss member with bottom bottom chord.

3.4 Check for Buckling Failure

[Reference: Cl 7.1.2.1 IS 800: 2007]

$$f_{cd} = \frac{f_y / \gamma_{m0}}{\phi + \sqrt{\phi^2 - \lambda^2}} \quad (3.6)$$

where,

$$\phi = 0.5[1 + \alpha(\lambda - 0.2) + \lambda^2]$$

λ = non-dimensional effective slenderness ratio;

$$= \sqrt{f_y / f_{cc}}$$

$$f_{cc} = \text{Euler Buckling stress} = \frac{\pi^2 E}{(K l_c / r_{min})^2}$$

$K = 0.7$, Effective length factor;

l_c = Whitmore equivalent column length;

r_{min} = radius of gyration of plate = $t_g / \sqrt{12}$;

f_{cd} = Design Compressive stress;

t_g = Thickness of gusset plate.

$$\frac{F}{B_{eff} t_g} \leq f_{cd} \quad (3.7)$$

where,

F = Factored Axial force on truss member;

B_{eff} = Whitmore effective width.

Check 4

Check for Truss Member

4.1 Block failure check

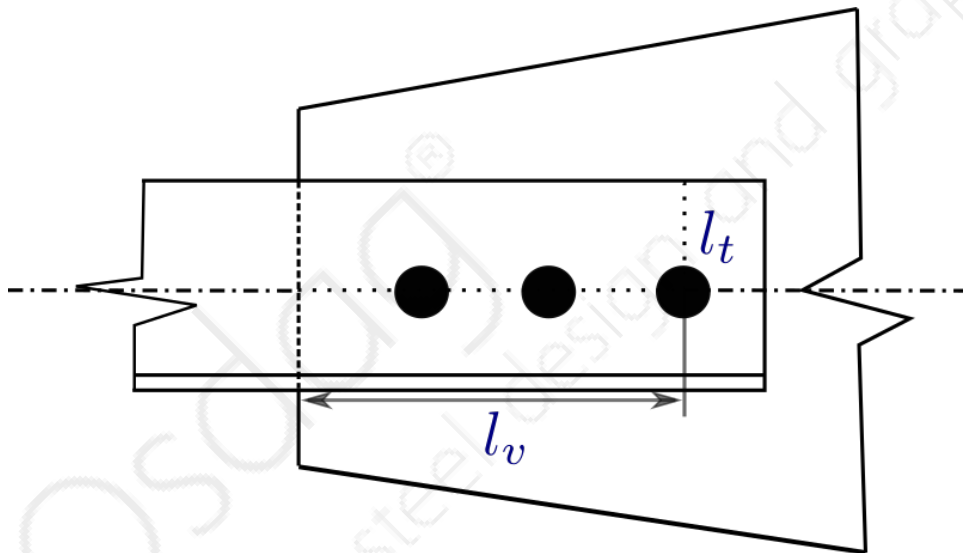


Figure 4.1: Block Failure of Truss member

$$T_{db1} = \frac{A_{vg}f_y}{\sqrt{3}\gamma_{m0}} + \frac{0.9A_{tn}f_u}{\gamma_{m1}} \quad (4.1)$$

$$T_{db2} = \frac{0.9A_{vn}f_u}{\sqrt{3}\gamma_{m1}} + \frac{A_{tg}f_y}{\gamma_{m0}} \quad (4.2)$$

where,

A_{vg} = Minimum gross area in shear along bolt line parallel to external force = $l_v * t$

A_{vn} = Minimum net area in shear along bolt line parallel to external force = $A_{vg} - (n_r - 0.5)d_0$

A_{tg} = Minimum gross area in tension from the bolt hole to the toe of the angle, end bolt line, perpendicular to the line of force = $l_t * t$

A_{tn} = Minimum net area in tension from the bolt hole to the toe of the angle, end bolt line, perpendicular to the line of force = $A_{tg} - (n_c - 0.5)d_0$

n_r = number of bolt rows

n_c = number of bolt columns

f_u = Ultimate stress of the plate material

f_y = Yield stress of the plate material

$\gamma_{m0} = 1.10$

$\gamma_{m1} = 1.25$

Check 5

Detailing

The size and shape of the gusset plates are decided based on the direction of various members meeting at a joint. The plate outlines are fixed so as to meet the minimum edge distance specified for the bolts that are used to connect various members at a particular joint and also it should give aesthetic appearance.

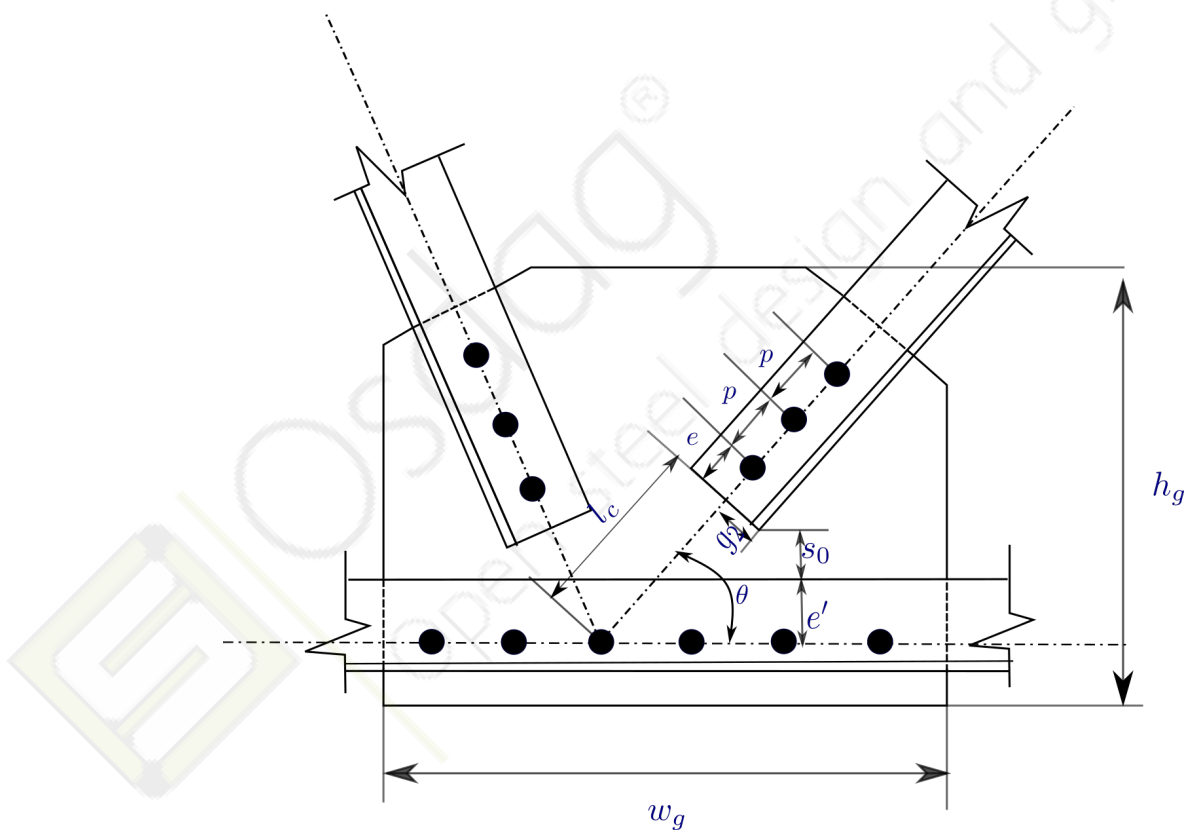


Figure 5.1: Gusset Plate detailing

5.1 Pitch (p) and Gauge (g)

[Reference: Cl. 10.2.2 and 10.2.3, IS 800 : 2007]

$$2.5 d \leq p \text{ or } g \leq \min(32 t, 300 \text{ mm}) \quad (5.1)$$

Where,

d = diameter of bolt;

t = thickness of the thinner plate.

5.2 End (e) and Edge (e')

[Reference: Clause 10.2.4, IS 800 : 2007]

$$[1.5 \text{ or } 1.7] \times d_0 \leq e/e' \leq 12 t \varepsilon \quad (5.2)$$

$$\varepsilon = \sqrt{\frac{250}{f_y}} \quad (5.3)$$

Where,

1.5 for machine cut and 1.7 for hand cut;

d_0 = diameter of the hole;

t = thickness of the thinner plate;

f_y = yield stress of the plate.

5.3 Gusset plate Dimensions

5.3.1 Height of Gusset plate (h_g)

$$h_g = \max[(l_c + l_{con} + 2e)\sin(\theta)] \quad (5.4)$$

5.3.2 Width of gusset plate (w_g)

5.3.2.1 case 1

When truss connection has no member in $\theta_{max} < 90^\circ$ or $\theta_{min} > 90^\circ$, width of gusset plate is given by;

$$L_1 = \max[(l_{con} + l_c + 2e)\cos(\theta)] \quad (5.5)$$

$$L_2 = p(n - 1) + e \quad (5.6)$$

$$w_g = L_1 + L_2 \quad (5.7)$$

5.3.2.2 case 2

When truss connection has members in $\theta < 90^\circ$ and $\theta > 90^\circ$, width of guesst plate is given by;

5.3.2.3 For ($\theta < 90^\circ$)

$$L_1 = \max[(l_{con_i} + l_{c_i} + 2e)\cos(\theta)] \quad (5.8)$$

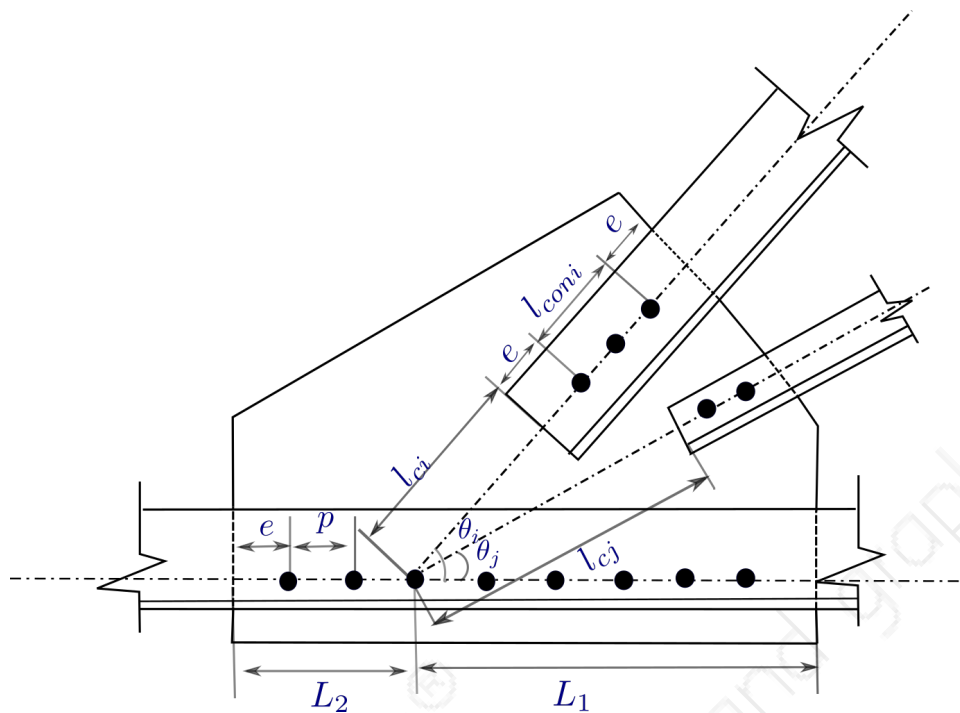


Figure 5.2: Width of Gusset Plate

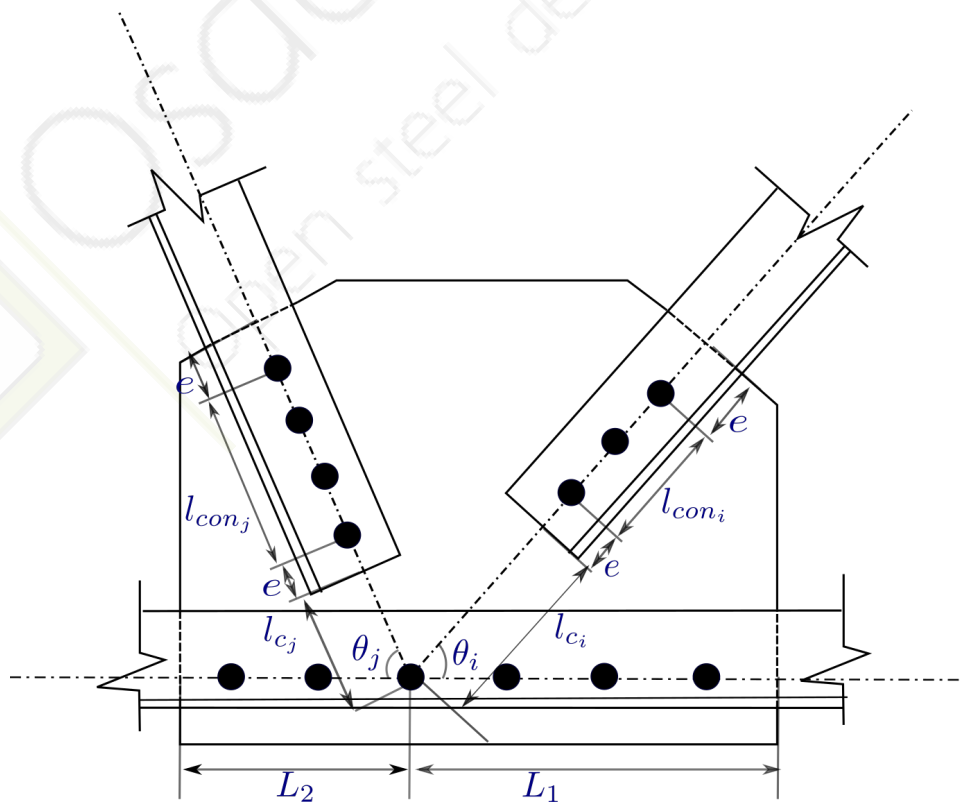


Figure 5.3: Width of Gusset Plate

5.3.2.4 For ($\theta > 90^\circ$)

$$L_2 = \max(l_{con,j} + l_{c,j} + 2e)\cos(\theta) \quad (5.9)$$

$$w_g = L_1 + L_2 \quad (5.10)$$

where,

n = number of bolts in bottom chord.

l_{con} = connection length of truss members;

l_c = Whitmore equivalent column length;

e = end distance for truss member;

p = pitch for the truss member;

n = number of bolts;

θ_{max} = angle of inclination of diagonal or vertical truss member with C.G of bottom chord.

Appendix G

DDCL for Welded Truss Connection



Design and Detailing Checklist (DDCL)
and
Design and Detailing Query (DDQ)

Welded Truss Connection

Prepared by:

Rachna Gupta

Under the guidance of:

Prof. Siddhartha Ghosh



Indian Institute of Technology, Bombay

July 13, 2019

Contents

1	Weld Design	6
1.1	Gusset plate thickness (t_g)	6
1.2	Weld Length of Truss Members(l_w)	6
2	Gusset Plate Checks	8
2.1	Whitmore Effective Width (B_{eff})	8
2.2	Check for Tension	8
2.2.1	Check for Tension Yielding	8
2.3	Compression Check	9
2.3.1	Whitmore equivalent column length (l_c)	9
2.4	Check for Buckling Failure	9
2.5	Block Failure of Gusset Plate	9
3	Detailing	11
3.1	Gusset plate Dimensions	11
3.1.1	Height of Gusset plate (h_g)	11
3.1.2	Width of gusset plate (w_g)	11

Reviewer Details & Guideline(s) for filling DDCL/DDQ

1. Name of the reviewer:

2. Institute/Company/Organization:

3. Designation:

- This document is for Design and Detailing Check List (DDCL) and Design and Detailing Query (DDQ) created by Osdag team, IIT Bombay.
- The checks are documented algorithmically and chapter wise in the document where the checks and sub-checks are given in the section and subsection respectively.
- Reference of the check is given just below the check title
- Each check has an associated checkbox and a comment box for giving feedback. The checkbox can be checked
- If you check on the 'Not OK' checkbox during the review, please specify your reason in the comment box with reference(s) (if any). It is mandatory!
- Send the document after review to sgshosh@civil.iitb.ac.in.

For any queries or help on filling the feedback document, contact Rachna Gupta at [rachna291998@gmail.com]

Reviewer Details



User Inputs

- Connecting members
Truss Member Sections*
- Connectivity*
Welded
- Material Property
 f_u (MPa)*
 f_y (MPa)*
- Factored loads
Axial Force (kN)
- Detailing
Angle of inclination(degree)*
- Plate
Thickness (mm)*
 f_u (MPa)*
 f_y (MPa)*
- Weld size
weld size (mm)*

Design and Detailing Checks



Osdag®

Open steel design and graphics

Check 1

Weld Design

Truss members and gusset plates are connected by weld

1.1 Gusset plate thickness (t_g)

Thickness of the gusset plate provided is usually equal to or slightly higher than members that are connected to gusset plate.

Thickness of gusset plate is calculated on the basis of maximum force acting on the truss members and is given by the table. [Reference: Table 5.12 Design of steel structure by N.Subramanian]

Maximum design force in diagonal(KN)	upto 200	200-450	450-750	750-1650	1650-2250	2250-3000
Thickness of gusset plate(mm)	8	10	12	16	18	20

1.2 Weld Length of Truss Members(l_w)

[Reference: Page no. 484 Design of steel structure by N.Subramanian Edition 13th]

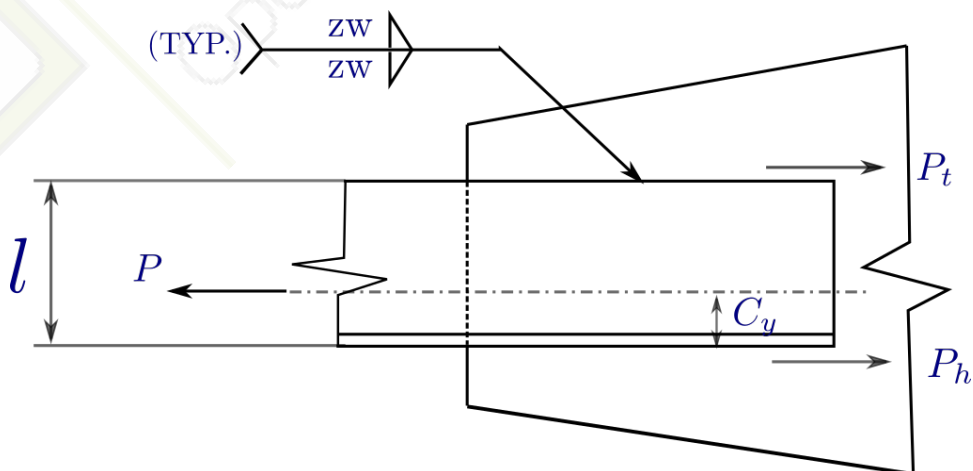


Figure 1.1: Balancing the weld on a tension member connection

$$f_w = \frac{t_t f_u}{\sqrt{3} \gamma_{m1}} \quad (1.1)$$

$$P_t = \frac{F C_y}{l} \quad (1.2)$$

$$P_h = F - P_t \quad (1.3)$$

where,

f_w = Strength of weld;

F = Axial Force on truss member;

P_h = Axial force resisted by weld in hill side;

P_t = Axial force resisted by weld in toe side;

C_y = C.G of bottom chord;

t_t = Throat thickness of weld = $0.7s$;

s = weld size;

f_u = minimum of ultimate strength of truss member and welded material;

γ_{m1} = Partial safety factor for ultimate strength;

l = leg size of truss member section.

$$l_{wt} = \frac{P_t}{f_w} \quad (1.4)$$

$$l_{wh} = \frac{P_h}{f_w} \quad (1.5)$$

where,

l_{wt} = Overall length of weld in toe side;

l_{wh} = Overall length of weld in hill side;

$$l_{wteff} = l_{wt} - 2s \quad (1.6)$$

$$l_{wheff} = l_{wh} - 2s \quad (1.7)$$

where,

l_{wteff} = Effective length of weld in toe side;

l_{wheff} = Effective length of weld in hill side.

Check 2

Gusset Plate Checks

2.1 Whitmore Effective Width (B_{eff})

[Reference: 5.12 Truss Joint Connection Page no. 365 Design of steel structure by N.Subramanian Edition 13th]

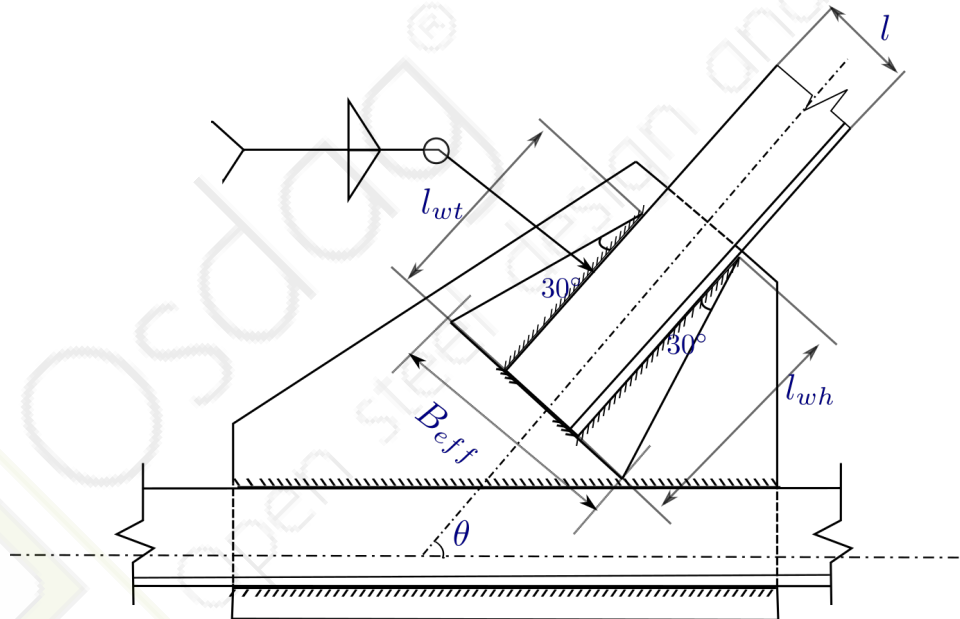


Figure 2.1: Effective Width of Gusset Plate as per Whitmore

$$B_{eff} = l + l_{wt}\tan(30^\circ) + l_{wh}\tan(30^\circ) \quad (2.1)$$

2.2 Check for Tension

2.2.1 Check for Tension Yielding

[Reference: Cl 6.2 IS 800: 2007]

$$\frac{P}{(B_{eff}t_g)} \leq \frac{f_y}{\gamma_{m0}} \quad (2.2)$$

where,

P = Axial force on each member;

B_{eff} = Whitmore effective width;
 t_g = Gusset plate thickness;
 f_y = yield strength of plate;
 γ_{m0} = Partial factor of safety for yielding.

2.3 Compression Check

2.3.1 Whitmore equivalent column length (l_c)

[Reference: Design of steel by structure N. Subramanian]

$$l_c = \frac{s_0 + g_1 + g_2 \cos(\theta)}{\sin(\theta)} \quad (2.3)$$

where,

s_0 = spacing between bottom chord and the connected truss member;
 g_1 = gauge length of bottom chord;
 g_2 = gauge length of connected truss member;
 θ = angle of inclination of connected truss member with bottom bottom chord.

2.4 Check for Buckling Failure

[Reference: Cl 7.1.2.1 IS 800: 2007]

$$f_{cd} = \frac{f_y / \gamma_{m0}}{\phi + \sqrt{\phi^2 - \lambda^2}} \quad (2.4)$$

where,

$\phi = 0.5[1 + \alpha(\lambda - 0.2) + \lambda^2]$
 λ = non-dimensional effective slenderness ratio;
 $= \sqrt{f_y / f_{cc}}$
 f_{cc} = Euler Buckling stress = $\frac{\pi^2 E}{(K l_c / r_{min})^2}$
 $K = 0.7$, Effective length factor;
 l_c = Whitmore equivalent column length;
 r_{min} = radius of gyration of plate = $t_g / \sqrt{12}$;

f_{cd} = Design Compressive stress;
 t_g = Thickness of gusset plate.

$$\frac{F}{B_{eff} t_g} \leq f_{cd} \quad (2.5)$$

where,

F = Axial force on truss member;
 B_{eff} = Whitmore effective width.

2.5 Block Failure of Gusset Plate

[Reference: Cl 6.4.1 IS 800: 2007]

In case of welded connection gusset plate may fail by block shear may occur.

$$T_{db1} = \frac{A_{vg} f_y}{\sqrt{3} \gamma_{m0}} + \frac{0.9 A_{tn} f_u}{\gamma_{m1}} \quad (2.6)$$

$$T_{db2} = \frac{0.9A_{vn}f_u}{\sqrt{3}\gamma_{m1}} + \frac{A_{tg}f_y}{\gamma_{m0}} \quad (2.7)$$

where,

A_{vg} = Minimum gross area in shear along bolt line parallel to external force = $l_v * t$

A_{vn} = Minimum net area in shear along bolt line parallel to external force = $l_v * t$

A_{tg} = Minimum gross area in tension from the bolt hole to the toe of the angle, end bolt line, perpendicular to the line of force = $l_t * t$

A_{tn} = Minimum net area in tension from the bolt hole to the toe of the angle, end bolt line, perpendicular to the line of force = $l_t * t$

f_u = Ultimate stress of the plate material

f_y = Yield stress of the plate material

$\gamma_{m0} = 1.10$

$\gamma_{m1} = 1.25$

Check 3

Detailing

3.1 Gusset plate Dimensions

3.1.1 Height of Gusset plate (h_g)

Height of gusset plate is based on maximum weld length and angle of inclination of truss member with C.G of bottom chord.

$$h_g = \max(l_c + l_w) \sin(\theta) \quad (3.1)$$

where,

l_w = weld length of diagonal or vertical truss member;

l_c = Whitmore equivalent column length;

θ angle of inclination of truss member with C.G of bottom chord.

3.1.2 Width of gusset plate (w_g)

Width of gusset plate is equal to weld length of bottom chord.

$$w_g = L_{wb} \quad (3.2)$$

where,

L_{wb} =Length of weld for bottom chord.