

Politechnika Warszawska
Wydział Mechaniczny Energetyki i Lotnictwa



Obliczenia inżynierskie w chmurze
Program liczący zmęczeniowy współczynnik
bezpieczeństwa - sprawozdanie

Szymon Kramarczyk

313636

Warszawa

25.01.2025r.

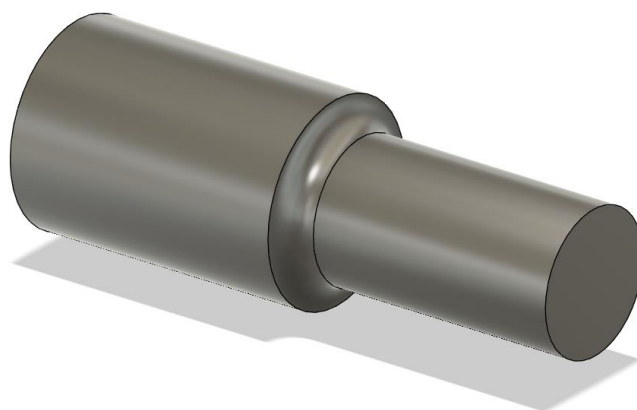
1. Opis projektu

Celem projektu było opracowanie narzędzia umożliwiającego aproksymację współczynników mechanicznych na podstawie danych tabelarycznych za pomocą metod interpolacji numerycznej. W projekcie wykorzystano dwuwymiarową interpolację regularną do oszacowania wartości współczynników potrzebnych do policzenia zmęczeniowego współczynnika bezpieczeństwa.

Projekt zakładał także integrację danych wczytywanych z plików tekstowych oraz uwzględnienie analizy danych pochodzących z pliku .json, z możliwością łatwej modyfikacji i rozszerzenia ich zawartości. Realizacja projektu wymagała zastosowania biblioteki scipy, w celu użycia zawartej w niej metody interpolacji RegularGridInterpolator.

2. Działanie programu

Program umożliwia aproksymację zmęczeniowego współczynnika bezpieczeństwa geometrii sworznia z osadzeniem o pewnym promieniu, wykonanym ze stali, na podstawie wartości wejściowych dostarczanych przez użytkownika. Przykładowa geometria, dla której przeprowadzone zostały obliczenia przedstawiona została poniżej:



Wartościami potrzebnymi do obliczeń, opisującymi powyższą część są:

- d - Mniejsza średnica sworznia
- D - Większa średnica sworznia
- r - Promień zaokrąglenia karbu
- R_e i R_m - Granica plastyczności i wytrzymałości doraźnej stali z której wykonany jest sworzeń

- Rodzaj naprężeń w sworzniu – jednostronnie odzerowo tętniące lub obustronnie symetryczne
- σ_{\max} - Maksymalne obciążenia (Von Misesa) zaobserwowane w analizie MES obiektu, po obciążeniu zadaną siłą
- Typ obróbki powierzchni materiału – toczenie zgrubne, szlifowanie etc.
- Stan materiału – ulepszony, surowy lub wyżarzony

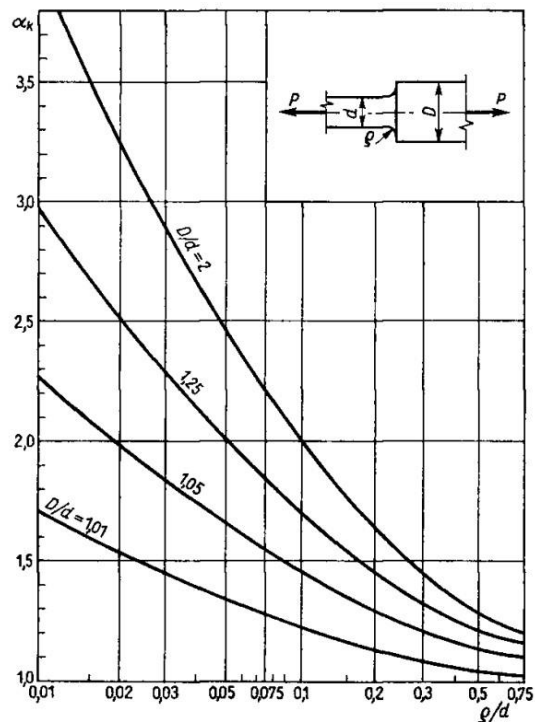
Do przykładowej analizy przyjęto następujące wartości (geometria sworznia mocującego siłownik hydrauliczny do otwierania stalowego wjazdu bunkra magazynowego):

- $d - 40\text{mm}$
- $D - 56\text{mm}$
- $r - 8\text{mm}$
- $R_e - 235\text{MPa}$
- $R_m - 360\text{MPa}$
- Obciążenia jednostronnie odzerowo tętniące o wartości maksymalnej $\sigma_{\max} = 76\text{MPa}$
- Obróbka zwykłym toczeniem
- Surowa stal konstrukcyjna

Dane te pozwoliły na policzenie współczynników i innych danych potrzebnych do dalszych obliczeń:

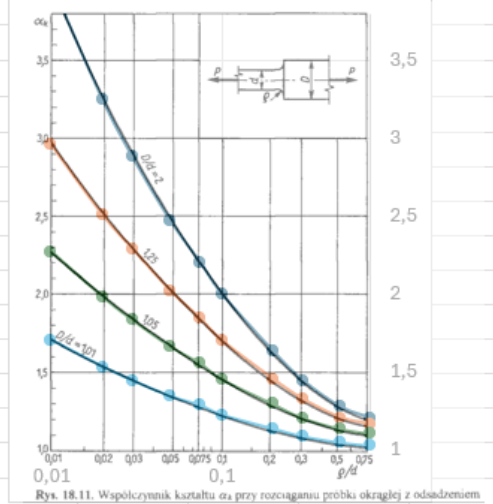
- Współczynnik kształtu α_k
- Materiałowy współczynnik wrażliwości na działanie karbu η
- Współczynnik wrażliwości na działanie karbu β_k
- Współczynnik stanu powierzchni β_p lub β_{ps}
- Współczynnik wielkości przedmiotu γ

Każda z tych wartości musi być odczytana z odpowiednich wykresów, ich wartości zmieniają się wraz z najmniejszymi zmianami w modelu oraz w założeniach, przez co ręczne liczenie zmęczeniowego współczynnika bezpieczeństwa potrafi być bardzo uciążliwe i podatne na błędy odczytów „na oko”.



Pierwszym krokiem pisania programu było stworzenie siatki punktów odpowiadającej każdemu z wykresów w programie Excel i eksportowanie danych do plików .txt

Rozciąganie próbki okrągłej z osadzeniem					
alpha k		D/d			
		2	1,25	1,05	1,01
r_o/d	0,01	4	2,96	2,27	1,7
	0,02	3,25	2,51	1,98	1,53
	0,03	2,88	2,29	1,83	1,44
	0,05	2,47	2,02	1,66	1,34
	0,075	2,2	1,84	1,55	1,28
	0,1	2	1,7	1,45	1,22
	0,2	1,63	1,45	1,29	1,13
	0,3	1,44	1,32	1,2	1,08
	0,5	1,27	1,2	1,13	1,04
	0,75	1,2	1,16	1,1	1,02



Rys. 18.11. Współczynnik kształtu α_k przy rozciąganiu próbki okrągłej z osadzeniem

Dane tabelaryczne następnie są wczytywane z plików tekstowych, które zawierają zarówno wartości zmiennych niezależnych, jak i odpowiadające im wartości współczynników z użyciem poniższej funkcji:

```
def load_table(file_path):
    with open(file_path, 'r') as f:
        lines = f.readlines()
    header_1 = list(map(float, lines[0].split())) # Pierwsza linia
    header_2 = list(map(float, lines[1].split())) # Druga linia
    table = np.array([list(map(float, line.split())) for line in lines[2:]] # Tabela
    return header_1, header_2, table
```

Program wykorzystuje funkcję RegularGridInterpolator do przeprowadzenia dwuwymiarowej interpolacji, co pozwala na uzyskanie precyzyjnych wyników w przypadku wartości spoza dostępnych danych tabelarycznych.

```
# Wczytywanie danych z plików
Zgo, stan_stali, eta_table = load_table('./notch.txt')

# Tworzenie interpolatorów
eta_interpolator = RegularGridInterpolator((Zgo, stan_stali), eta_table, bounds_error=False, fill_value=None)

# Funkcja obliczająca Współczynnik podatności na działanie karbu
def Notch_sens_factor(Zgo, stan_stali, eta_interpolator):
    point = np.array([Zgo, stan_stali])
    return eta_interpolator(point).item()
```

W dalszej części kodu wczytywane są dane początkowe na których podstawie liczone są wartości potrzebne do wykonania odczytów z wykresów:

<pre>#Wczytanie danych liczbowych z pliku json i obliczenie potrzebnych wartości config = load_config("./config.json") ro = config["ro"] d = config["d"] D = config["D"] Re = config["Re"] Rm = config["Rm"] sigma_max = config["sigma_max"] x = config["x"] load_type = config["load_type"] load_alternation_type = config["load_alternation_type"] typ_obrobki = config["typ_obrobki"] stan_stali = config["stan_stali"]</pre>	<pre>ro_d = ro/d D_d = D/d if load_type == 1: Zo = 0.5*Rm*0.7 Zj = 0.5*Rm*1.3 elif load_type == 2: Zo = 0.5*Rm Zj = 0.5*Rm*1.7 elif load_type == 3: Zo = 0.5*Rm*0.6 Zj = 0.5*Rm*1.2 if load_alternation_type == 1: sigma_a=sigma_max/2 sigma_m=sigma_max/2 elif load_alternation_type == 2: sigma_a=sigma_max sigma_m=0</pre>
--	--

Ostatnim krokiem jest wywołanie funkcji aproksymujących wartości współczynników i podstawienie ich do wzoru na zmęzeniowy współczynnik bezpieczeństwa:

```
alpha_k = Shape_factor(ro_d, D_d, alpha_values_load(load_type))
eta = Notch_sens_factor(Zo, stan_stali, eta_interpolator)
beta_k = 1+eta*(alpha_k-1)
beta_p = Surface_finish_factor(Rm, typ_obrobki, beta_p_values_load(load_type))
beta = beta_k*beta_p
gamma = Size_factor(Zo, alpha_k, d)

safety_coeff = Zo/((beta*gamma*sigma_a)+sigma_m*((2*Zo/Zj)-1))
```

3. Platforma Azure

Ostatnim krokiem projektu było stworzenie wirtualnej maszyny na platformie Azure, na której skompilowany będzie projekt. Jako że siatka punktów wykresów jest dosyć rzadka, (dane te są ekperymentalne i przez to mało dostępne) program w obecnej formie nie jest zbyt skomplikowany obliczeniowo. Z tego powodu zdecydowano się na użycie najtańszej wirtualnej maszyny, z systemem Ubuntu 24.04. Parametry komputera zostały przedstawione poniżej:

Running	Standard DS1 v2 (1 vcpu, 3.5 GiB memory)
Location	Public IP address
West Europe (Zone 2)	50.85.81.211
Subscription (move)	Virtual network/subnet
Azure for Students	szymon-vnet/default

```
szymon@szymon:~$ neofetch
      ,--/+00SSSS00+/- ,
      `:+SSSSSSSSSSSSSSSSSS+:`
      -+SSSSSSSSSSSSSSSSSSSSSSSS+-
      .0SSSSSSSSSSSSSSSSSSSSdMMMNySSSS0.
      /SSSSSSSSSSShdmmNNmmyNMMMMhSSSSSS/
      +SSSSSSSSShmydMMMMMMNNdddySSSSSSSS+
      /SSSSSSSSShNMMMyhyyyyhmNMMMNhSSSSSSS/
      .SSSSSSSSdMMMNhSSSSSSSSShNMMMdSSSSSSSS.
      +SSSSShhyNMMNySSSSSSSSSSSYNMMMySSSSSSSS+
      0SSyNMMMNyMMhSSSSSSSSSSShmmhSSSSSSSS0
      0SSyNMMMNyMMhSSSSSSSSSSShmmhSSSSSSSS0
      +SSSSShhyNMMNySSSSSSSSSSSYNMMMySSSSSSSS+
      .SSSSSSSSdMMMNhSSSSSSSSShNMMMdSSSSSSSS.
      /SSSSSSSSShNMMMyhyyyyhdNMMMNhSSSSSSS/
      +SSSSSSSSSdmydMMMMMMNNdddySSSSSSSS+
      /SSSSSSSSShdmmNNmmyNMMMMhSSSSSS/
      .0SSSSSSSSSSSSSSSSSSSSdMMMNySSSS0.
      -+SSSSSSSSSSSSSSSSSSSSSSSS+-
      `:+SSSSSSSSSSSSSSSSSS+:`
      ,--/+00SSSS00+/- ,

szymon@szymon:~$
```

```
szymon@szymon
-----
OS: Ubuntu 24.04.1 LTS x86_64
Host: Virtual Machine Hyper-V U
Kernel: 6.8.0-1020-azure
Uptime: 5 hours, 43 mins
Packages: 782 (dpkg)
Shell: bash 5.2.21
Resolution: 1024x768
Terminal: /dev/pts/0
CPU: Intel Xeon Platinum 8171M
Memory: 429MiB / 3358MiB
```

Następnie połączono się z maszyną wirtualną za pomocą protokołu SSH wykorzystując terminal maszyny macierzystej:

```
~ ssh szymon@50.85.81.211
The authenticity of host '50.85.81.211 (50.85.81.211)' can't be established.
ED25519 key fingerprint is SHA256:Uku66ykjSxLhffUIXByeV13PXngxQke8QzPdk1bXg
AM.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '50.85.81.211' (ED25519) to the list of known ho
sts.
szymon@50.85.81.211's password:
Welcome to Ubuntu 24.04.1 LTS (GNU/Linux 6.8.0-1020-azure x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/pro

System information as of Sun Jan 26 16:03:47 UTC 2025

System load:  0.68      Processes:    125
Usage of /:   5.4% of 28.02GB   Users logged in:  0
Memory usage: 6%      IPv4 address for eth0: 10.1.0.4
Swap usage:   6%

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

szymon@szymon:~$
szymon@szymon:~$
```

Kolejnym krokiem było zaktualizowanie repozytoriów oraz zainstalowanie Dockera poprzez komendy:

```
szymon@szymon:~$ sudo apt update
```

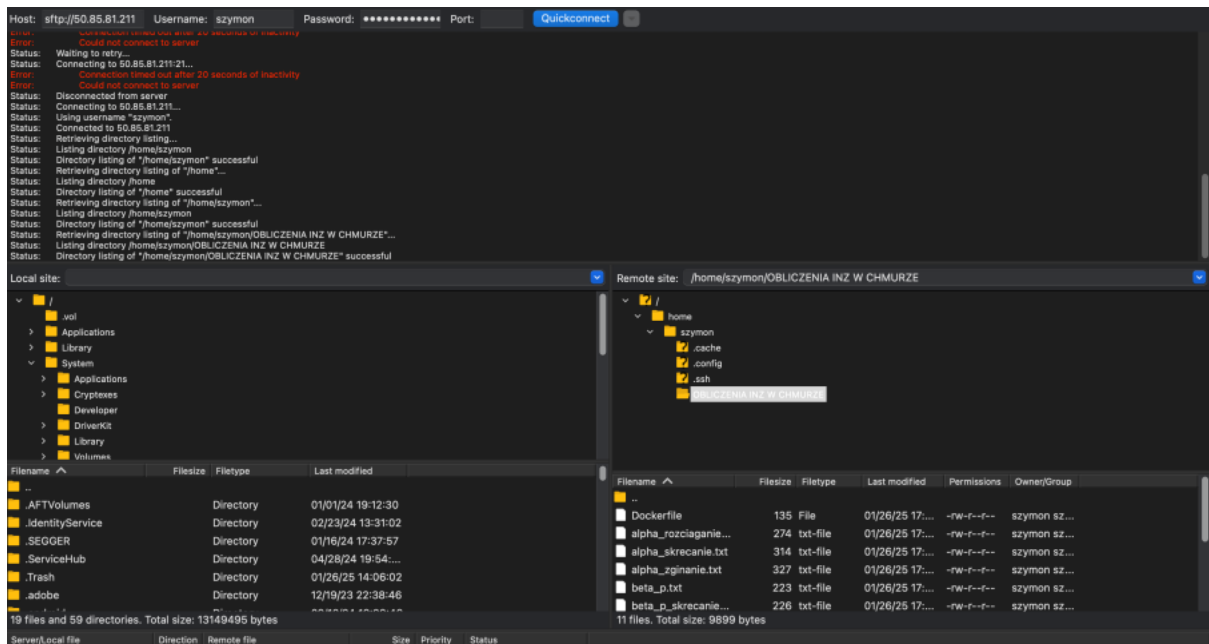
```
szymon@szymon:~$ sudo apt install docker.io
```

Po tym kroku można użyć programu Filezilla do wgrania plików programu na maszynę wirtualną. W tym kroku z początku napotkano pewne problemy z programem Filezilla, dlatego najpierw wgrano pliki z użyciem innego podejścia, używając komendy pokazanej poniżej:

```
szymon@szymon:~$ tree
.
├── OBLICZENIA INZ W CHMURZE
│   ├── Dockerfile
│   ├── alpha_rozciąganie.txt
│   ├── alpha_skrecanie.txt
│   ├── alpha_zginanie.txt
│   ├── beta_p.txt
│   ├── beta_p_skrecanie.txt
│   ├── config.json
│   ├── gamma_table.txt
│   ├── notch.txt
│   ├── wspolczynnik_bezpieczenstwa.py
│   └── x_table.txt
└── 2 directories, 11 files
```

```
Downloads scp -r OBLICZENIA\ INZ\ W\ CHMURZE szymon@50.85.81.211:/home/si
szymon
szymon@50.85.81.211's password:
beta_p_skrecanie.txt      100% 226   3.8KB/s  00:00
beta_p.txt                100% 223   3.7KB/s  00:00
Dockerfile               100% 135   2.3KB/s  00:00
config.json              100% 988  13.3KB/s 00:00
gamma_table.txt          100% 805  12.7KB/s 00:00
notch.txt                100% 177   2.9KB/s  00:00
alpha_rozciąganie.txt    100% 274   4.8KB/s  00:00
alpha_zginanie.txt       100% 327   5.7KB/s  00:00
wspolczynnik_bezpieczenstwa.py 100% 5976 46.8KB/s 00:00
alpha_skrecanie.txt      100% 314   5.2KB/s  00:00
x_table.txt              100% 454   4.5KB/s  00:00
```

W kolejnych krokach rozwiązano jednak problem i uzyskano połączenie w przewidywany na zajęciach sposób:



Ostatnimi krokami były odpowiednio zbudowanie obrazu na podstawie napisanego dockerfile oraz jego wykonanie przez komendę dockera:

```
szymon@szymon:~/OBLICZENIA INZ W CHMURZE$ sudo docker build -t python .
```

```
szymon@szymon:~/OBLICZENIA INZ W CHMURZE$ sudo docker run --name python -it python
Aproksymowany współczynnik kształtu: 1.41
Aproksymowany współczynnik jakości powierzchni: 1.10
Aproksymowany współczynnik wrażliwości na korb: 0.67
Aproksymowany współczynnik wielkości przedmiotu: 1.24

Obliczony zmęczeniowy współczynnik bezpieczeństwa: 2.47

szymon@szymon:~/OBLICZENIA INZ W CHMURZE$
```


4. Podsumowanie

Otrzymane wyniki zgadzają się z wynikami odczytywanymi „ręcznie”, a nawet pomogły znaleźć błąd w obliczeniach wspomnianego wcześniej sworznia mocującego siłownik. Poprawne ręczne obliczenia przedstawiono w tabeli poniżej:

	Re	Rm				Formula	FOR S235JR	FOR S355JR
S235JR	235	360		Shape factor	α_k	plot 18.12	1,40	1,40
S355JR	355	470		Material notch sensitivity factor	η	plot 18.9	0,68	0,73
				Notch sensitivity factor	β_k	$1 + \eta(\alpha_k - 1)$	1,27	1,29
Pin smaller diameter	d	40,00		Surface finish factor	β_p	plot 18.6	1,10	1,08
Pin larger diameter	D	56,00		Fatigue stress concentration factor	β	$\beta_k \cdot \beta_p$	1,40	1,40
Fillet radius	p	8,00		Size factor	Y	plot 18.1	1,25	1,25
	D/d	1,40		Fatigue strength under alternating bending	Zgo	0,5 * Rm	180	235
	p/d	0,20		Fatigue strength under unidirectional bending	Zgj	0,85 * Rm	306	399,5
Max stress	σ_{max}	76,00		Bending yield strength	Qz	1.2 * Re	282	426
Stress amplitude	σ_a	38		fatigue safety coefficient	xz	$x_z = \frac{Z_{go}}{\beta \gamma \sigma_a + \sigma_m} \geq x_{rw}$		
Mean stress	σ_m	38					2,46	3,22
Safety factor	x	2,00				$x_z = \frac{Q_z}{\beta \gamma \sigma_a + \sigma_m} \geq x_{rw}$		
							2,70	4,09

Aproksymacja jest wystarczająco dokładna na potrzeby tych obliczeń oraz upraszcza i co najważniejsze znacznie skraca proces obliczeń wytrzymałościowych analizowanego obiektu. Uniwersalna metoda wprowadzania danych pozwala również na przyszłe modyfikacje i dostosowanie programu do pomocy przy obliczeniach innych geometrii oraz materiałów a także do przyjęcia dokładniejszych danych do aproksymacji współczynników.