SVEUČILIŠTE U ZAGREBU
GRAĐEVINSKI FAKULTET

PREDNAPETE KONSTRUKCIJE OD UŽADI

ZAVRŠNI RAD IZ PREDMETA GRAĐEVNA STATIKA 2

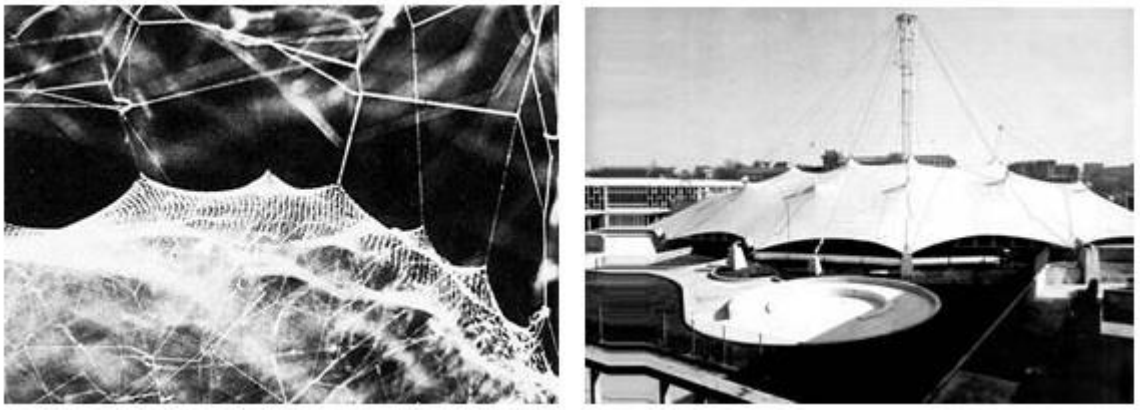
Student: Silvije Šajn, 0082045524
Mentor: prof. dr. sc. Krešimir Fresl

Zagreb, 17. rujan 2013.

Sadržaj:

1. O prednapetim konstrukcijama od užadi

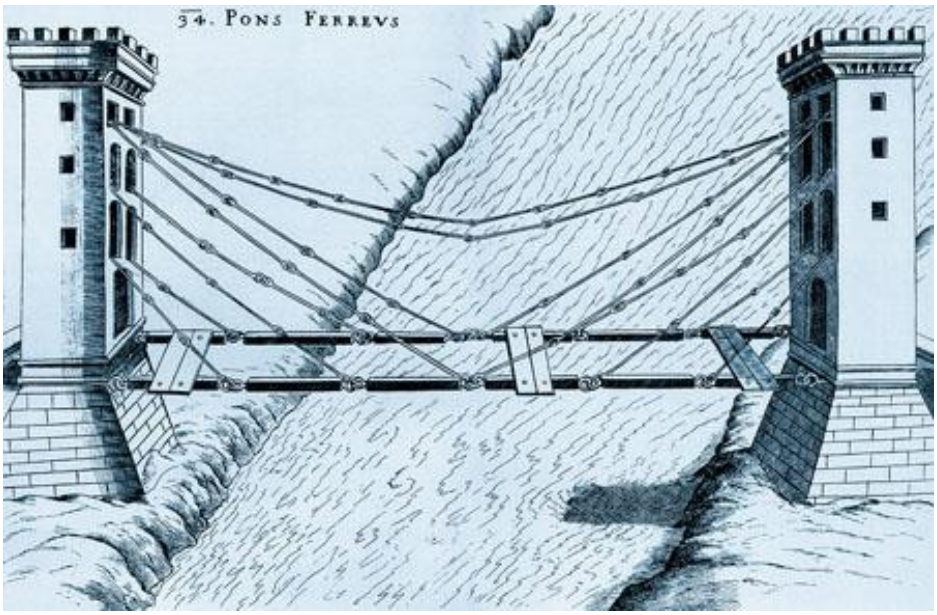
Projektiranje konstrukcija od užadi pripada složenim arhitektonskim i konstruktivnim problemima. Za izgradnju takvih konstrukcija trebalo je pronaći nove metode projektiranja, proračuna i analize, budući da se one bitno razlikuju od tradicionalnih konstrukcija. Rješenje tog problema pronađeno je upravo u prirodi (slika 1.), jer priroda uvijek troši minimum energije za ostvarivanje određene namjene. Prirodna struktura uvijek će poprimiti takav oblik pri kojem će potencijalna energija naprezanja unutar nje biti minimalna. Dok se nije pronašlo rješenje, konstrukcije su se u najranijoj primjeni izrađivale na temelju iskustva, bile su jednostavnijih oblika i radile su se od priručnih materijala.



1. Mreža pauka Cyrtophora i Boulevard Carnot Swimming Pool u Parizu

1.1. Povijest i razvoj

Jedna od prvih upotreba konstrukcija od užadi i platna bila je namijenjena za gradnju nastamba nomadskih naroda. Njihova mala težina, jedostavna i brza montaža i demontaža bile su neke od glavnih prednosti zbog kojih se pribjegavalo njihovoj uporabi. Tijekom prošlosti užad je upotrebljavana i kao nosivi element prvih mostova. Prvu zamisao izvedbe ovješenih mostova dao je polihistor Faust Vrančić u svojoj knjizi „*Machinae novae“* (slika 2.) Ona se smatra ishodištem upotrebe užadi za izgradnju ovješenih i visećih mostova. U opisu mosta stoji da je bio načinjen od dva brodska konopa ili više njih, zavješenih na dva stupa na obje obale. Da bi most stajao napet i da se ne bi savijao od težine onih koji preko njega prolaze, treba konope, zavezane za brodsku užad, prema potrebi zategnuti ili popustiti. Ovaj se most može sklopiti i prenositi, stoga dobro služi i u ratu.



2. Nacrt ovješenog mosta Fausta Vrančića

Najveći razvoj vlačnih građevina za natkrivanje prostora odvijao se u posljednjih 60-ak godina,a prvi su se tom problematikom bavili njemački inženjeri Frei Otto i Jörg Schlaich. Pionirom u projektiranju lakih građevina smatra se Frei Otto, arhitekt, inženjer i znanstvenik koji je svoja djela radio prema uzorima iz prirodnih stanja.

Za razliku od ranijih šatorastih konstrukcija Otto je u Montrealu primijenio novi tip membrane koja je složena od jasno diferenciranih elemenata - nosive čelične mreže i translucentne opne, koncepcija koju će dalje rabiti za pokrivanje glavnih sportskih borilišta Olimpijade u Münchenu 1972. Njemački paviljon za Expo 1967. u Montrealu natkriven je jedinstvenim šatorastim krovom, nepravilna oblika i različitih visina, koji je pokrivao ukupno 7700 m^2 tlocrtne površine (slika 3.). Nosiva prednapeta mreža od čeličnih kabela razapeta je preko osam čeličnih jarbola, visine od 14 do 38 metara, i po rubu fiksirana za sidrene točke na tlu. Ottova višegodišnja nastojanja za pronalaženjem superlagane i ekonomične konstrukcije uporabom novih materijala i tehnologija rezultirala su elegantnom uglačanom membranom, istovremeno vizualno napetom i mekanom, što je arhitekturu oslobodila tradicionalnih oblika evocirajući arhetipski oblik šatora starih nomadskih plemena. Ingeniozno rješenje zadivljuje još više ako se zna da je paviljon, proizveden i dopremljen u dijelovima iz Njemačke, montiralo samo 25 radnika u dva mjeseca.

3. Njemački paviljon u Montrealu, Frei Otto

1.2. Današnja primjena

Zbog male specifične mase tih konstrukcija, primjenjuje ih se za natkrivanje velikih raspona kao što su stadioni, športske dvorane ili izložbeni paviljoni, gdje se zahtijeva da veliki prostor bude neprekinut elementima konstrukcije. U tehnološkom smislu izvedba je brza i svodi se na montažu tvornički prefabriciranih elemenata, što uz malu težinu te konstrukcije čini gotovo jedinstveno mobilnima.

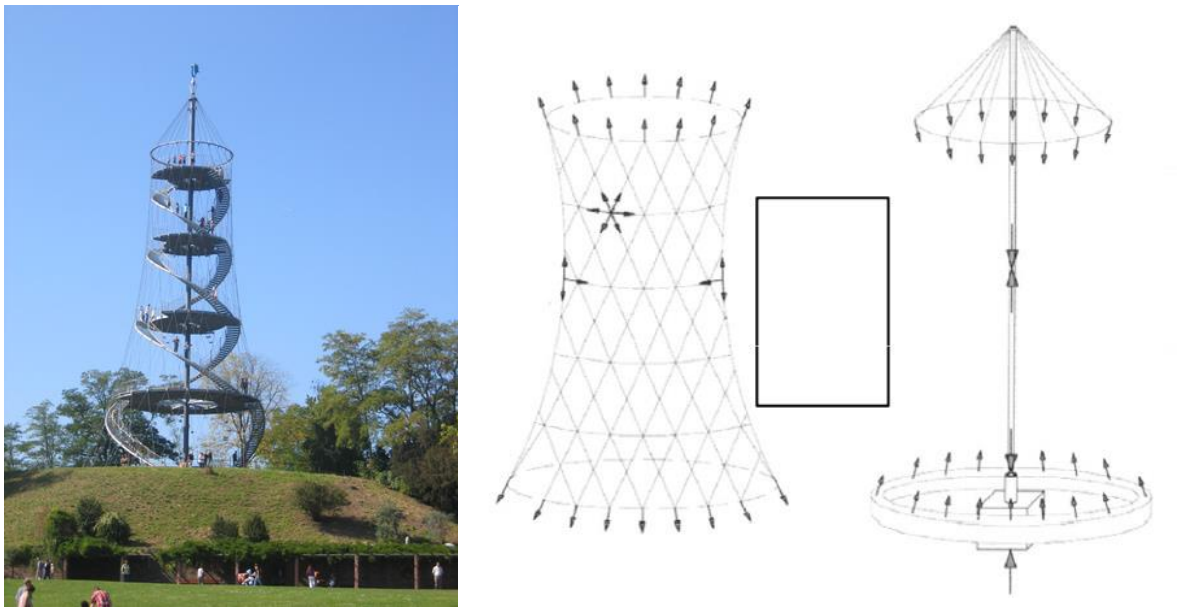
Prednapeta užad sastavni je dio ovješenih i visećih mostova, ona je element pomoću kojeg su ostvareni veliki rasponi tih mostova. Tu valja nadodati da se svakodnevno razvijaju novi materijali, boljih karakteristika, veće otpornosti kojima je ostvarena evolucija vlačnih konstrukcije u prava arhitektonska i projektantska djela.

Osim za krovove i mostove užad se rabi i u druge svrhe. Užad je upotrijebljena kao nosivi element kod raznih tornjeva. Jedan od takvih primjera je i rashladni toranj od mreže kabela atomske centrale u Schmehausenu pokraj Hamma u Njemačkoj (slika 4.).

4. Rashladni toranj atomske centrale od mreže kabela, Schmehausen

Prednapeta mreža kabela oblika rotacijskog hiperboloida je podignuta na središnji jarbol od betona, a gore i dolje su tlačni prstenovi sandučastog čeličnog presjeka. Kabeli su postupno dodavani podizanjem svakog segmenta, a u konačnici je s unutarnje strane postavljena obloga. Toranj je srušen 1991.

Još jedan je primjer toranj visine 40,3 m koji služi kao vidikovac, a sastoji se od dvostrukog spiralnog stubišta i platformi koje su obješene na mrežu od 48 kabela (slika 5.). Užad je na vrhu pričvršćena na srednji tlačni jarbol, a oblik formiraju čelični tlačni prsten na vrhu i kružni temelj na dnu gdje je užad usidrena.



5. Toranj *Killesberg,* Stuttgart (2001)

1.3. Razlika u odnosu na tradicionalne konstrukcije

Na temelju sile kojom prenose opterećenja, prednapete konstrukcije od užadi svrstavamo u vlačne konstrukcije, a njih zbog optimalne uporabe materijala svrstavamo u lagane konstrukcije. Njihova se učinkovitost očituje u tome što se za prijenos opterećenja koriste vlačnim silama, za razliku od ostalih tradicionalnih oblika koji sile prenose tlakom, savijanjem i torzijom. Posljedica toga je manji utrošak materijala za izvedbu takvih konstrukcija, jer se materijal rabi optimalno samo ako je pod utjecajem membranskih sila, a ne momenata savijanja. Vlačni se elementi mogu opterećivati sve do granice vlačne čvrstoće materijala, te su tako materijal i poprečni presjek potpuno iskorišteni. Zbog male mase i velike čvrstoće pogodne su za velike raspone kod mostova i natkrivanja velikih površina bez dodatnih oslonaca i ležajeva u sredini raspona. Međutim zbog male težine vlačne konstrukcije potrebni su elementi koji će se odupirati odizućem opterećenju (najčešće od vjetra).

Zbog velike mogućnosti oblikovanja i estetskih vrijednosti ističu se i svojom elegancijom i atraktivnošću. Zakrivljenost vlačnih konstrukcija osigurava otpornost na utjecaj vjetra ili pokretnoga opterećenja koje djeluje na plohu konstrukcije. Velika prednost u odnosu na konvencionalne konstrukcije su mogućnosti lake montaže i demontaže, reciklaže, te transporta.

Kod tradicionalne konstrukcije od betona, čelika i drva stabilnost i prijenos sila im omogućuju vlastita težina i krutost, dok je za razliku od njih, težina vlačnih struktura gotovo zanemariva (prosječno iznosi 0,1kN/m^2), a materijal iz kojega su izrađene je izrazito fleksibilan. Samim time stabilnost i nosivost moraju se postići na drugačiji način. Elementi koji tvore vlačnu konstrukciju moraju formirati specifični geometrijski oblik i biti pod utjecajem određenih unutarnjih sila.

Postoje velike sličnosti između vlačne membrane i prednapete mreže kabela. Općenito se može reći da se platno rasteže više nego kabel, a kabel više nego kruti konstrukcijski element.

1.4. Podjela

Pravčasta mreža

Pravčasta mreža je mreža u kojoj svako uže leži na pravcu tj. užad čije su osi pravci na pravčastoj plohi ili prostornoj pravčastoj mreži (slika 6.). Njezin se oblik može odrediti geometrijski , bez postupka nalaženja oblika, a ravnoteža je ostvarena budući da je prednaponska sila konstantna uzduž svakog užeta bez međudjelovanja s ostalima, premda se s nekima od njih dodiruje. Kod nje ne postoji opasnost od gubitka prednapona pod djelovanjem poprečnog opterećenja. Odsječak pravca je najkraća moguća spojnica između dvije čvrste točke u prostoru, pa se uže pričvršćeno između dvije čvrste točke zbog poprečnih pomaka može samo produljiti i još više nategnuti. Mlohavost štapova se onemogućava minimalnim prednapinjanjem. Konstrukcije bi tada zbog malih uzdužnih sila i zbog pravčaste geometrije bile još gipkije od ostalih mrežastih konstrukcija, što zbog velike deformacije pokrova i dinamičke osjetljivosti izaziva teškoće u upotrebi. Zbog toga se ne primjenjuje u građevinarstvu.

One se opisuju izrazom:

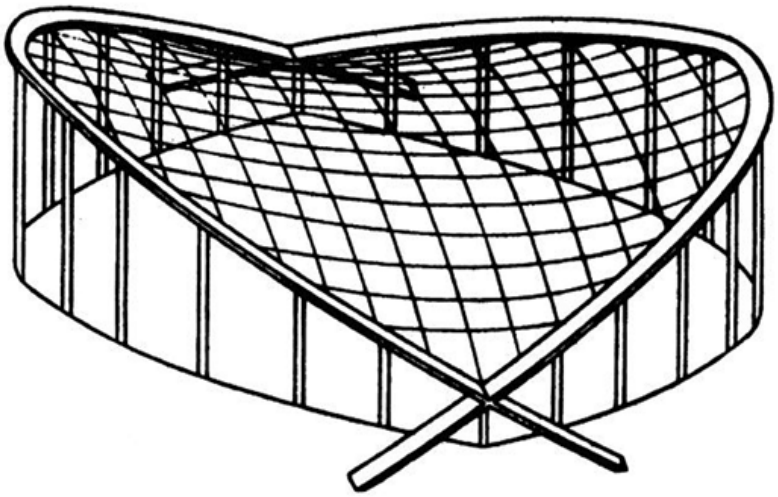
$$x(u,v) = l(u) + vk(u)\ 0 \le u,v \le 1$$

gdje je *x* pravčasti segment plohe.

6. Pravčasta mreža u kružnom tlocrtu

Pravilna ili regularna mreža

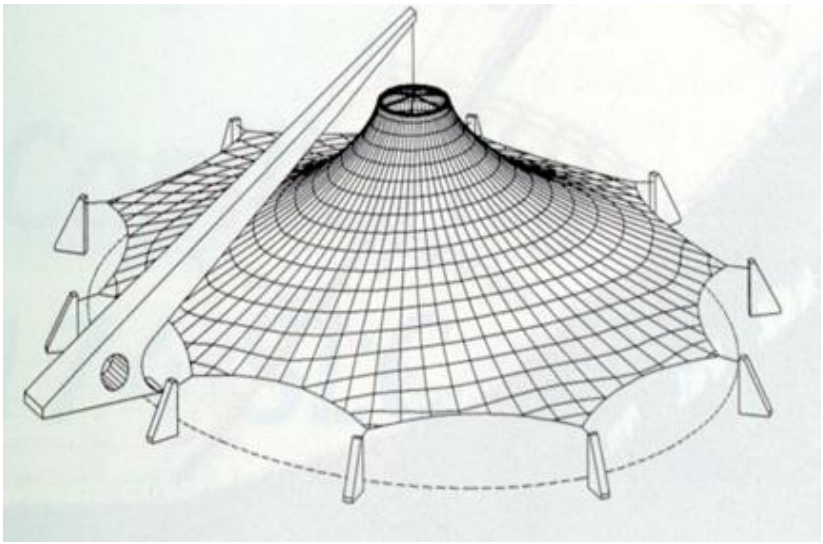
Sastoji se od dvije familije užadi koje međudjeluju i tvore sedlasti oblik (slika 7.). Pri djelovanju vertikalnog opterećenja orijentirana prema dolje, te se sile u konkavnoj (nosivoj) užadi povećavaju, dok se u konveksnoj (prednaposnkoj) užadi smanjuju.



7. Regularna mreža

Neregularna mreža

Neregularna mreža sastoje se više od dviju familija užadi (slika 8.). Kod nje uže može biti na jednom djelu konkavno, a na drugom konveksno, zatim može biti prekinuto u nekom čvoru ili se unutar mreže može nalaziti neki kruti element. Samim time ona je složenijeg oblika od regularne mreže.



8. Neregularna mreža

1.5. Redosljed projektiranja konstrukcija od užadi

Oblikovanje i projektiranje vlačnih konstrukcija je vrlo složen postupak i u mnogo čemu se razlikuje od projektiranja konstrukcija od betona, drva i čelika. Osim što zahtijeva dobro razumijevanje ponašanja užadi i njihovih konstrukcija, bitno je i dobro poznavanje računalnog programa koji primjenjujemo. Zbog nelinearnosti proračuna, metode elementarne mehanike ne mogu se rabiti kod proračuna.

Projektiranje se može podijeliti na sljedećih 5 postupaka:

a) Odluka o sistemu koji ćemo primjeniti

Budući da je užad efikasna za konstrukcije velikih raspona, neki sistemi su pogodniji za određene arhitektonske oblike i zahtjeve. Užad također ovisi o fizičkim svojstvima materijala, rasponu za koji se namjenjuje, kvalifikaciji i stručnosti radne snage i o cijeni.

b) Postupak nalaženja oblika

Zadatak tog procesa je naći oblik konstrukcije čija geometrija zadovoljava uvjete ravnoteže i oblika. O tom problemu ćemo govoriti u *2. poglavlju.*

c) Analiza konstrukcije

Danas se analiza provodi pomoću softvera koji na temelju ulaznih podataka provodi statičku i dinamičku analizu konstrukcije.

d) „Krojenje“ veličina elemenata i priključaka

Duljina kabela je odabrana u procesu nalaženja oblika i analize konstrukcije i teži se da bude minimalna. Kabeli također moraju zadovoljiti u pogledu omjera nosivosti i mase .

Budući da je $l_{i,j} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2}$, duljina jednog odsečka kabela između dva čvora, ukupna duljina jednaka je sumi svih odsječaka

$$L = \sum_{(i,j)} l_{i,j} = \sum_{(i,j)} \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2} \ .$$

Minimalnu duljinu kabela dobivamo deriviranjem duljine *L* po svim komponentama *x, y, z* i izjednačavanjem s 0, jer kad je duljina minimalna, tad uže u tom položaju ima i minimalnu energiju:

$$\frac{\partial L}{\partial x_i} = \sum_{j=1}^{n} \frac{x_i - x_j}{\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2}} = 0 \ ,$$

$$\frac{\partial L}{\partial y_i} = \sum_{j=1}^{n} \frac{y_i - y_j}{\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2}} = 0 \ ,$$

$$\frac{\partial L}{\partial z_i} = \sum_{j=1}^{n} \frac{z_i - z_j}{\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2}} = 0 \ .$$

Priključci se upotrebljavaju za spajanje dva križno postavljena kabela da se spriječi njihovo međusobno klizanje, zatim priključci za krajeve kabela kojima se oni spajaju u sidrene elemenete ili okvirnu konstrukciju.

e) Gradnja vlačne konstrukcije

Budući da je već u prethodnim procesima detaljno opisana konstrukcija, sad je na redu slaganje kabela u mrežu i njihovo prednapinjanje. Problem se javlja kod redosljeda prednapinjanja kablova, jer su oni sad međuzavisni. Napinjanje jednog kabela može izazvati vlačna naprezanja u nekom drugom, zato se kod analize treba pronaći i optimalan redosljed da se u konačnosti ne bi javile velike razlike prednapona u mreži.

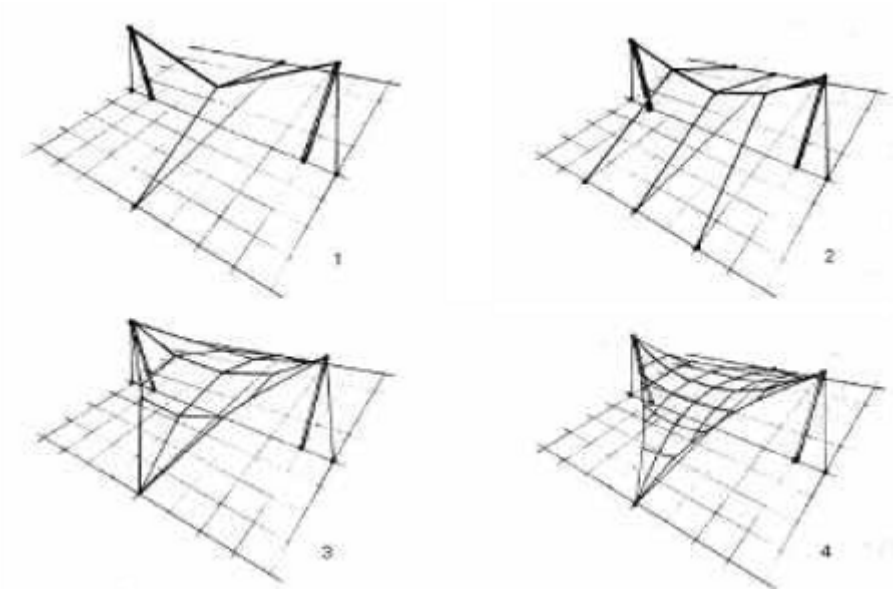
2. Problem nalaženja oblika

2.1. O obliku vlačnih konstrukcija

Već je ranije spomenuto da se oblik vlačnih konstrukcija ne može nametnuti kao što je to slučaj kod tradicionalnih konstrukcija, već se mora naći ravnotežan oblik vlačne konstrukcije za odabrani raspored prednaponskih sila i ležajnih točaka. Naći oblik znači odrediti ravnotežno stanje tj. uravnotežiti prednaponske sile i vanjsko opterećenje.

Prednapete mreže kabela i platnene membrane karakterizira interakcija između njihove geometrije i raspodjele naprezanja. Upravo zbog te veze između oblika i sila nemoguće je projektirati vlačne konstrukcije izravno, kao što je to slučaj s konvencionalnim konstrukcijama.

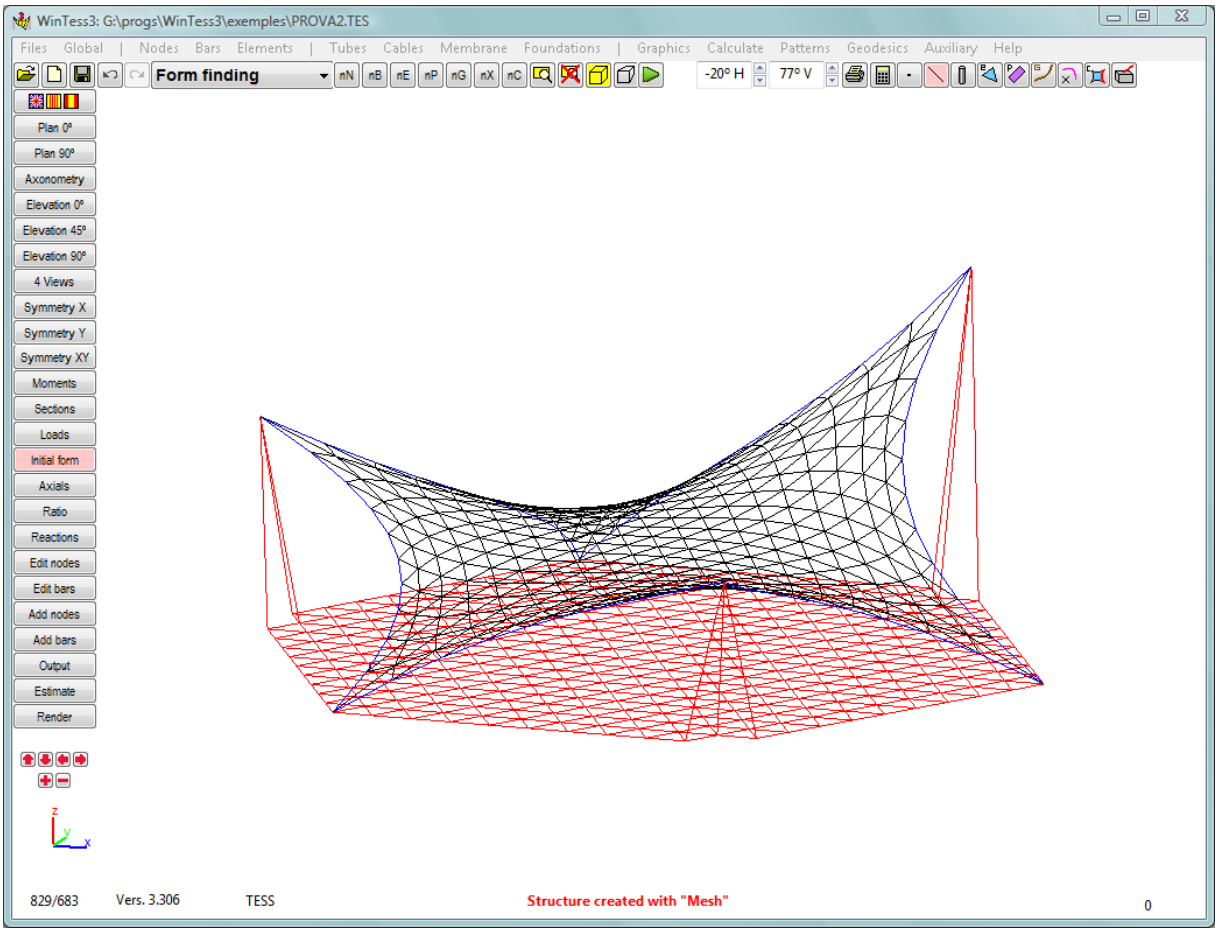
Vlačna konstrukcija zahtijeva minimalno četiri ležajne točke (slika 9.), a jedna od četiri točke mora biti izvan ravnine koju određuju ostale tri kako bi ploha razapeta između te četiri točke bila sedlasta, jer upravo dvostruka zakrivljenost daje konstrukciji stabilnost i nosivost.



9. Sedlasta ploha razapeta između četiri točke

Odabirom ležajnih mjesta definira se i oblik konstrukcije. Geometrija ležajnih točaka zajedno s definiranim prednaponom vodi ka određenom ravnotežnom obliku. Kada se zadaju rubni uvjeti konstrukcije i distribucija unutarnjih, prednaponskih sila, tada postoji samo jedna trodimenzionalna ploha koja je u ravnoteži pod zadanim uvjetima (u svakoj točki).

Oblik te plohe određuje se matematičkim postupkom zvanim traženje oblika (engl. *form - finding*). Ako se želi dobiveni oblik izmijeniti, sile prednapona ili mjesta ležaja moraju biti promijenjena. Vidimo da mora postojati uska suradnja arhitekata i inženjera. Traženjem oblika želi se naći ravnotežni oblik vlačne konstrukcije i to za odabrani raspored prednaponskih sila. Taj dio proračuna karakterističan je upravo za vlačne konstrukcije (slika 10.).



10. Postupak nalaženja oblika na računalu

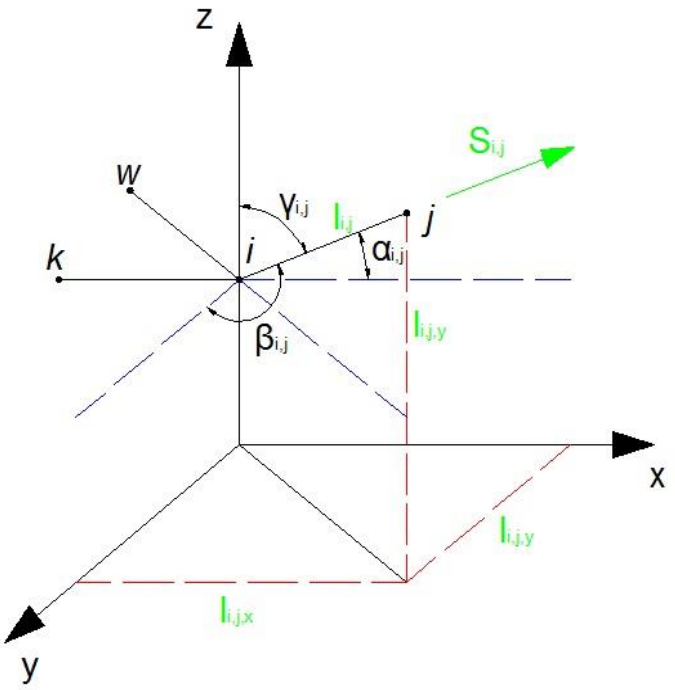
Složenost postupka oblikovanja proizlazi iz činjenice da je njihov oblik definiran mehaničkim načelima na koje projektant može samo posredno utjecati. Stoga se arhitektonska idejna studija ponajprije razrađuje pomoću fizičkih modela izrađenih na temelju osnovnih skica. Unaprijed zadovoljavajući zakonitosti prirode, oni osiguravaju idealnu predodžbu njihove često zamršene prostorne geometrije te čine osnovu na temelju koje će inženjer konstruktor u sljedećoj etapi procesa, uz pomoć računala, konstrukciji dati optimalan oblik. Ako se konstrukcija izradi od gipkog materijala (užadi, tkanine) koji ne može prenijeti tlačne sile ni momente savijanja, ona će pod opterećenjem zauzeti položaj koji zadovoljava statičke uvjete ravnoteže. Njezin će oblik ovisiti o rubnim uvjetima i odnosu prednaponskih sila. Time projektant, mijenjajući rubne uvjete, kreira oblik.

Prednapon je potreban zbog stabilnosti i nosivosti konstrukcije, a može se unositi kotvama na krajevima kabela. Inicijalni prednapon mora biti na otprilike 1/20 vlačne čvrstoće materijala, jer druga promjenjiva i stalna opterećenja imaju puno veće intenzitete djelovanja, samim time mora se osigurati da užad ostane u području elastičnih deformacija. Oblik i prednapon moraju osigurati vlak kroz čitav vijek trajanja građevine, samim time dajući konstrukciji takozvanu geometrijsku krutost. Velike prednaponske sile učinit će mrežu krućom i smanjiti deformaciju, ali će se istodobno povećati trošak izgradnje. Krutost prednapetih mreža kabela ovisit će o zakrivljenosti kabela, poprečnom presjeku kabela, nivou prednapona i krutosti potporne konstrukcije.

2.2. Uvjeti ravnoteže čvora bez vanjskog opterećenja

Simetrični zapis u kojemu su sve koordinate ravnopravne

Kao proračunski model mreže uzimamo jedan čvor u koji su priključeni kabeli. Kabele između dva čvora nazivamo štapovima i pretpostavljamo da se zglobno spajaju u čvorove.



Zbrojevi projekcija svih sila prednapona u štapovima priključenima u čvor na osi koordinatnog sustava moraju biti u ravnoteži s komponentama vanjskog opterećenja:

$$\sum_{j=1}^{n} X_{i,j} = 0 \ ,$$

$$\sum_{j=1}^{n} Y_{i,j} = 0 \ , \qquad (2.2.1.)$$

$$\sum_{j=1}^{n} Z_{i,j} = 0 \ ,$$

gdje je je *n* broj užadi pričvršćen u i-ti čvor.

Promotrimo štap između čvorova *i* i *j.* Ako kutove koje uzdužna sila u štapu zatvara s koordinatnim osima označimo sa $\alpha_{i,j}$, $\beta_{i,j}$ i $\gamma_{i,j}$, tada jednadžbe (2.2.1.) možemo zapisati u obliku:

$$\sum_{j=1}^{n} S_{i,j} \cdot \cos\alpha_{i,j} = 0 \ ,$$
$$\sum_{j=1}^{n} S_{i,j} \cdot \cos\beta_{i,j} = 0 \ , \qquad (2.2.2.)$$
$$\sum_{j=1}^{n} S_{i,j} \cdot \cos\gamma_{i,j} = 0 \ .$$

Budući da kosinusi $\alpha_{i,j}$,$\beta_{i,j}$ i $\gamma_{i,j}$ iznose:

$$\cos\alpha_{i,j} = \frac{l_{i,j,x}}{l_{i,j}} = \frac{x_i - x_j}{l_{i,j}} = \frac{x_i - x_j}{\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2}} \ ,$$
$$\cos\beta_{i,j} = \frac{l_{i,j,x}}{l_{i,j}} = \frac{y_i - y_j}{l_{i,j}} = \frac{y_i - y_j}{\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2}} \ , \qquad (2.2.3.)$$
$$\cos\gamma_{i,j} = \frac{l_{i,j,x}}{l_{i,j}} = \frac{z_i - z_j}{l_{i,j}} = \frac{z_i - z_j}{\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2}} \ ,$$

gdje su:

$l_{i,j} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2}$ - duljina štapa *i,j* ,

$l_{i,j,x}$, $l_{i,j,y}$, $l_{i,j,z}$ - duljine projekcija štapa u smjeru koordinatnih osi x,y,z ,

x_i, y_i, z_i – koordinate čvora *i* u prostoru,

x_j, y_j, z_j – koordinate čvora *j* u prostoru,

Jednadžba (2.2.2.) poprima sljedeći oblik:

$$\sum_{j=1}^{n} \frac{S_{i,j} \cdot (x_i - x_j)}{\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2}} = \sum_{j=1}^{n} \frac{S_{i,j} \cdot (x_i - x_j)}{l_{i,j}} = 0 \ ,$$

$$\sum_{j=1}^{n} \frac{S_{i,j} \cdot (y_i - y_j)}{\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2}} = \sum_{j=1}^{n} \frac{S_{i,j} \cdot (y_i - y_j)}{l_{i,j}} = 0 \ , \qquad (2.2.4.)$$

$$\sum_{j=1}^{n} \frac{S_{i,j} \cdot (z_i - z_j)}{\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2}} = \sum_{j=1}^{n} \frac{S_{i,j} \cdot (z_i - z_j)}{l_{i,j}} = 0 \ .$$

Gornje jednadžbe (2.2.4) predstavljaju konačni oblik zapisa jednadžbi ravnoteže čvora *i.*

Konačni oblik mreže određen je trima koordinatama x, y, z svakog čvora u kojemu se sijeku dva kabela, što ukupno čini *3n* nepoznanica, pri čemu je *n* broj slobodnih čvorova. Za svaki se čvor mogu napisati tri uvjeta ravnoteže u kojima se kao nepoznanice pojavljuju još i sile u pripadajućim štapovima. Konačno se dobiva sustav od *3n* jednadžbi s *3n+m* nepoznanica (gdje je *m* ukupni broj elemenata mreže) čije rješenje nije jednoznačno. Da bi sustav postao rješiv, potrebno je dodatno uvesti neke pretpostavke i ograničenja (ukupno *m* nepoznanica) koja će dovesti do jedinstvenog rješenja, ali i uvjetovati oblik konstrukcije.

Ako dobiveni oblik ne zadovoljava, mogu se zadati i drukčije pretpostavke, npr.:

- zadane su sile prednaprezanja u svim štapovima – *poopćeno pravilo geodetske mreže,*
- unutarnja geometrija mreže, odnosno udaljenosti među čvorovima ili smjer kabela,
- u nekim štapovima zadani su iznosi sila, dok je u ostalima definirana linearna ovisnost sile o deformaciji – *kombinirano pravilo,*
- zadane su uravnotežene veličine projekcija sila na ravninu xy– *Kvazilaplaceovo pravilo,*
- zadan je odnos sile i duljine pojedinih elemenata – *pravilo gustoća sila.*

Oblik minimalne mreže definira se određivanjem koordinata svih čvorova mreže pomoću izraza (2.2.4.). Ti izrazi čine sustav nelinearnih jednadžbi (kod poopćenog pravila geodetske mreže i kombiniranog pravila sustav jednadžbi ostaje nelinearan) koji se rješava jednom od poznatih numeričkih metoda, npr. Newton-Raphsonovom metodom. Nepoznanice se ne mogu izračunati izravno, već se na temelju pretpostavljenoga aproksimativnog rješenja (početno rješenje) algoaritam iterativnim postupkom približava konačnom rješenju.

Kasnije je potrebno ispitati zadovoljava li oblik dobiven jednim od tih ograničenja zahtjeve arhitekta i inženjera konstruktora, te ako je potrebno, postupak ponoviti, što zajedno čini proces optimizacije konstrukcije. Ako se do zadovoljavajućeg rješenja ne može doći, može se pribjeći i promjeni geometrije rubova.

U dosadašnjem tekstu se spominjnjao problem nalaženja oblika prednapetih konstrukcija samo pod djelovanjem sila prednapona. Za konstrukcije koje osim sila prednapona imaju znatno vanjsko opterećenje, ono se ponekad mora uvažiti već u fazi nalaženja oblika. Takve konstrukcije ne moraju imati negativnu Gaussovu zakrivljenost. Neke od primjera tih konstrukcija su konstrukcije opterećene balastom i napuhane konstrukcije kod kojih se treba uzeti još i unutarnji pritisak.

3. Metoda gustoća sila

Metoda gustoća sila razvijena je za potrebe kompjutorskog modeliranja krova Olimpijskog stadiona u Münchenu. Za proračun takvog opsega dotad upotrebljavani fizikalni modeli nisu bili dovoljni za određivanje oblika. Napravljen je računalni program za rješavanje te zadaće utemeljen na metodi gustoće sila. Metoda primjenjuje linearan sustav jednadžbi za postavljanje uvjeta ravnoteže prednapetog užeta. Ova transformacija nelinearnoga problema u linearan omogućava direktno rješavanje. Metodom gustoća sila dobivaju se oblici vlačnih konstrukcija koji su u statičkoj ravnoteži.

Metode kojima se dobivaju početna rješenja za numeričko određivanje oblika minimalnih mreža moraju zadovoljiti zahtjeve jednostavnosti i brzine postupka, a pritom dati rezultate bliske konačnima kako bi kasniji numerički postupak rješavanja nelinearnih jednadžbi što brže konvergirao.

Osnovna zamisao metode gustoća sila jest da se jednadžbe ravnoteže (2.2.4.) lineariziraju.

To se postiže uvođenjem omjera nazvanog gustoćom sila:

$$\frac{S_{i,j}}{l_{i,j}} = q_{i,j} \ , \quad (3.1)$$

koji se uvrštava u jednadžbe ravnoteže (2.2.4.).

Jednadžbe(2.2.4.) sada primaju oblik:

$$\begin{aligned} &\sum_{j=1}^{n} q_{i,j} \cdot (x_i - x_j) = 0 \ , \\ &\sum_{j=1}^{n} q_{i,j} \cdot (y_i - y_j) = 0 \ , \qquad (3.2) \\ &\sum_{j=1}^{n} q_{i,j} \cdot (z_i - z_j) = 0 \ . \end{aligned}$$

Time se dobivaju tri neovisna sustava čija se rješenja lako nalaze ako se kao početna pretpostavka uzme da su vrijednosti q_{ij} konstantne. Tako dobivena rješenja za jednostavne su slučajeve dobra aproksimacija. Jednim se sustavom određuju sve koordinate x_i, drugim sve koordinate y_i, a trećim sve koordinate z_i.

U pojedinim primjerima ta rješenja ne zadovoljavaju, pa se iz dobivenih rezultata računaju nove vrijednosti $q_{i,j}$ prema (3.1.). Time se dobivaju točniji, "popravljeni" rezultati, a postupak se može ponoviti u više iteracija.

4. Rješavanje sustava nelinearnih jednadžbi

Metodom gustoće sile i kvazilaplaceovim pravilom postižemo da nelinarni sustav jednadžbi ravnoteže (2.2.4.) postane linearan, dok kod poopćenog pravila geodetske mreže i kombiniranog pravila sustav jednadžbi ostaje nelinearan.

Jedna od metoda rješavanja sustava nelinearnih algebarskih jednadžbi je Newton - Raphsonova metoda ili metoda tangente. Ako početnu aproksimaciju izaberemo dovoljno blizu rješenja, onda metoda vrlo brzo konvergira prema rješenju sustava jednadžbi.

Najprije ćemo objasniti rješavanje *jednadžbe s jednom nepoznanicom*.

Želimo riješiti jednadžbu tj. pronaći *x* za koji vrijedi:

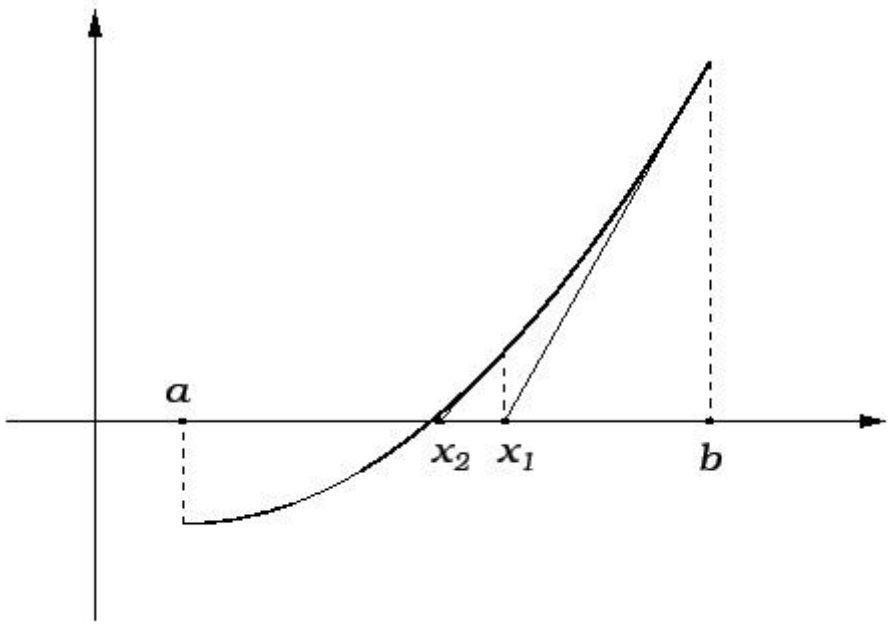
$$f(x) = 0 \ ,$$

gdje je:

f - neprekidna funkcija definirana na zatvorenom intervalu *[a, b]*.

Svaki kompleksan broj ε koji je rješenje jednadžbe nazivamo nul-točkom funkcije *f.*

Newtonova metoda se sastoji u tome da se (*n+1)- va* aproksimacija $x_{(n+1)}$ odredi kao sjecište tangente na graf funkcije *f* u točki s apscisom x_n s osi



x.

Izaberemo početnu aproksimaciju $x_n \in [a, b]$.

Zatim početnu funkciju f aproksimiramo linearnom funkcijom:

$$y = f(x_n) + f'(x_n)(x - x_n) ,$$

čiji je graf tangenta na graf funkcije f u točki $(x_n, f(x_n))$.

Sada umjesto jednadžbe f(x)=0 rješavamo jednadžbu y=0 (geometrijski gledano, umjesto traženja sjecišta grafa funkcije f s osi x, tražimo sjecište tangente s osi x), te dobijemo njezino sjecište s osi x, a to je (n+1)- va aproksimacija $x_{(n+1)}$.

Dakle,

$$0 = f(x_n) + f'(x_n)(x - x_n).$$

Odatle slijedi da je nul točka funkcije y :

$$x_{(n+1)} = x_n - \frac{(f(x_n))}{f'(x_n)}.$$

Iteracija je završena kada:

$$|f(x_n)| < \tau \text{ ili } n > N ,$$

gdje su:

τ-odabrana točnost,

N-odabran najveći broj koraka u slučaju divergencije.

Primjena Newtonove metode na rješavanje *sustava jednadžbi*:

$$f_i(x_1, x_2, x_3, ..., x_n) = 0 , \qquad i = 1,2,3,...,n,$$

odnosno $f(x) = 0$ gdje je $f: \mathbb{R}^n \to \mathbb{R}^n, f(x) = (f_1(x), ..., f_n(x))^T, x = (x_1, ..., x_n)^T$.

Najprije izaberemo početnu aproksimaciju

$$x^{(0)} = (x_1^{(0)}, x_2^{(0)}, ..., x_n^{(0)})^T \ ,$$

i svakoj od funkcija f_i linearnu aproksimaciju označimo s $\tilde{f}_i$

$$\tilde{f}_i = \tilde{f}_i(x^{(0)}) + \sum_{j=1}^{n} \frac{\partial f_i(x^{(0)})}{\partial x_j}(x_j - x_j^{(0)}) \ , \quad i=1,2,3,...,n,$$

Sada umjesto sustava $\mathrm{f}_i\left(\mathrm{x}_1, \mathrm{x}_2, \mathrm{x}_3, ..., \mathrm{x}_n\right) = 0$, $i=1,...,n,$ rješavamo sustav:

$$\tilde{f}_i(x) = 0 \ , \quad i=1,2,3,...,n,$$

koji u matričnom obliku izgleda:

$$J^{(0)} \cdot s^{(0)} = -f\left(x^{(0)}\right) \ ,$$

odnosno:

$$\begin{bmatrix} \frac{\partial f_1(x^{(0)})}{\partial x_1} & \cdots & \frac{\partial f_1(x^{(0)})}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_n(x^{(0)})}{\partial x_1} & \cdots & \frac{\partial f_n(x^{(0)})}{\partial x_n} \end{bmatrix} \cdot \begin{bmatrix} x_1 - x_1^{(0)} \\ \vdots \\ x_n - x_n^{(0)} \end{bmatrix} = \begin{bmatrix} f_1(x^{(0)}) \\ \vdots \\ f_n(x^{(0)}) \end{bmatrix}.$$

Nova aproksimacija rješenja sada je :

$$x^{(1)} = x^{(0)} + s^{(0)} \ .$$

Općenito, dobivamo iterativni postupak:

$$x^{(k+1)} = x^{(k)} + s^{(k)} \quad k=0,1,...,n,$$

gdje je vektor smjera $s^{(k)}$ rješenje sustava.

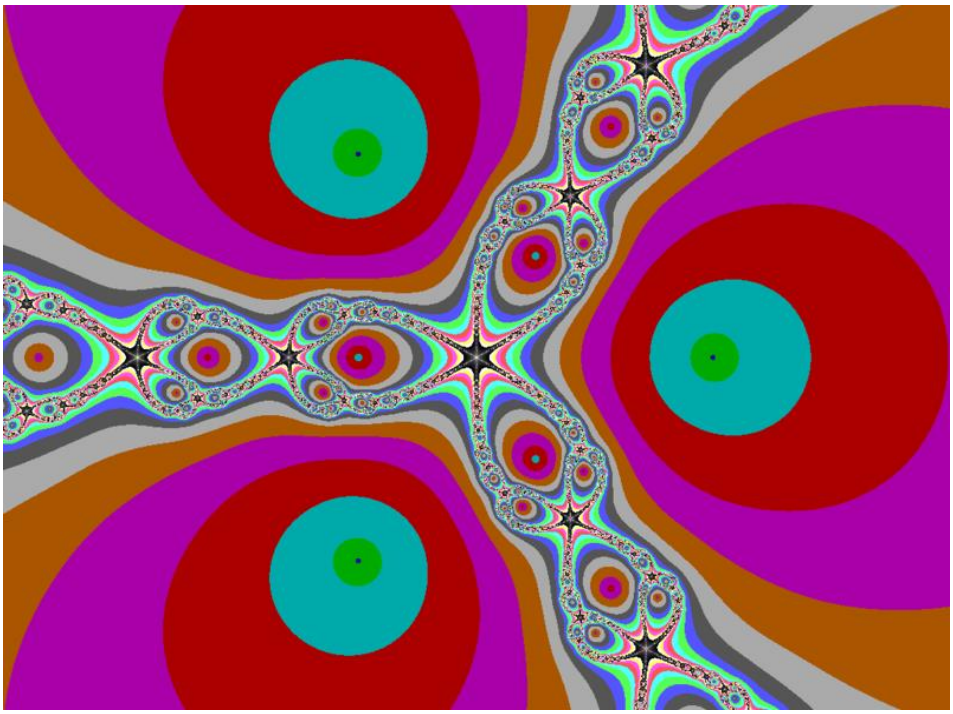
Nelinearni se sustav može rješavati iterativno koristeći se samo dijagonalnim elementima matrice *J*. Takva metoda rješavanja poznata je kao Newton-Gauss-Seidelova metoda rješavanja sustava nelinearnih jednadžbi:

$$x_i^{(k+1)} = x_i^{(k)} - \frac{f_i(x_1^{(k+1)}, ..., x_{i-1}^{(k+1)}, x_i^{(k)}, x_{i+1}^{(k)}, x_n^{(k)})}{\partial_{x_i} f_i(x_1^{(k+1)}, ..., x_{i-1}^{(k+1)}, x_i^{(k)}, x_{i+1}^{(k)}, x_n^{(k)})},$$

gdje je

$$\partial_{x_i} f_i = \frac{\partial f_i}{\partial x_i}.$$

Područje konvergencije ima izrazito nepravilan oblik, ponekad iteracija započela od neke odabrane aproksimacije konvergira pravom rješenju manjim brojem iteracija od neke prividno bolje aproksimacije. Varirajući početnu aproksimaciju uočavamo nepredvidivo ponašanje iteracijskog postupka, pa se došlo do pretpostavke da područje konvergencije ima svojstva fraktala (slika 11.).



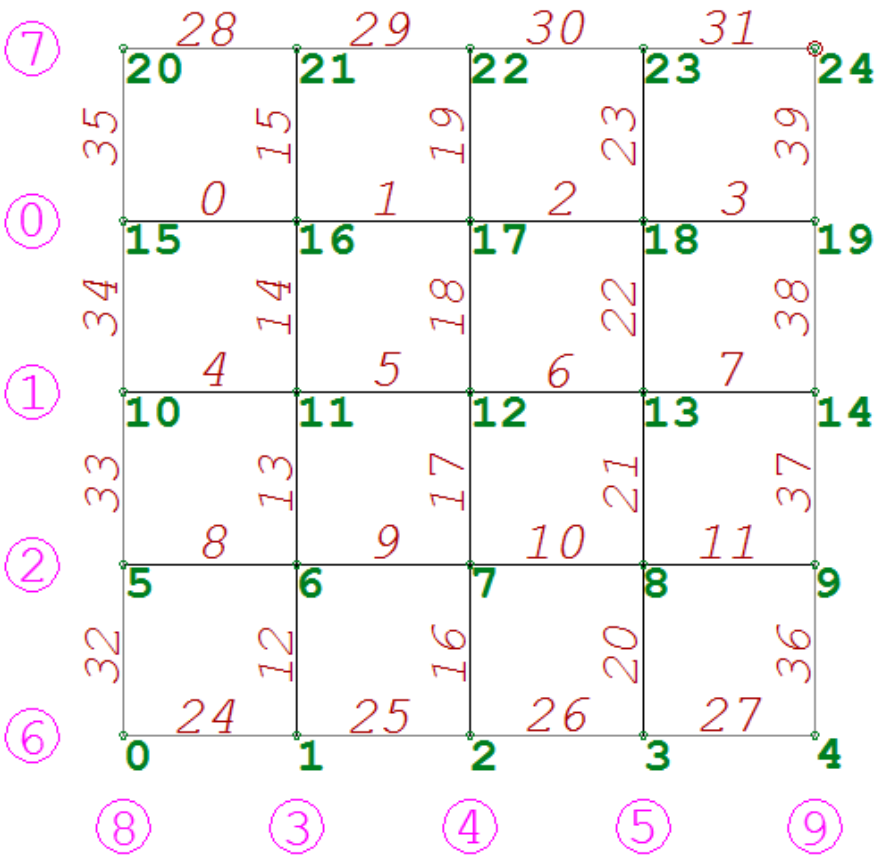
11. Newtonov fraktal za jednadžbu $p(z)=z^3-1$ obojeno ovisno o broju iteracija

5. Primjeri

Primjeri su riješeni primjenom programske funkcije newton_ krylov, sadržane u programskom paketu SciPy koji je uključen u simbolički programski paket *Sage*. Ta je funkcija programska realizacija Newton-Krilovljeva postupka rješavanja sustava nelinearnih jednadžbi, odnosno nalaženja nul-točke vektorske funkcije vektorske varijable. Newton-Krilovljev postupak inačica je poznatoga Newton-Raphsonova postupka. U svakom se koraku vanjske, Newton-Raphsonove petlje linearni sustavi rješavaju odabranim Krilovljevim iteracijskim postupkom.

Primjer 1.

Tražimo stabilni oblik mreže kabela koji se nalaži u ravnoteži. Mreža se sastoji od 10 *[0-9]* kabela koji se sastoje od 40 *[0-39]* štapova i ukupno 25 *[0-24]* čvorova. U mreži imamo 4 *[6-9]* rubna kabela na kojima se nalazi 16 *[24-39]* rubnih štapova. Ostalih 24 *[0-23]* štapova nalazi se na unutarnih 6 *[0-5]* kabela. Konstrukcija ima 4 ležajna čvora, od kojih 3 *[0, 4, 28]* leže u istoj ravnini, a jedan *[24]* je izvan nje.



Najprije zadajemo čvorove mreže:

- ovdje smo mogli svaku točku mreže upisati pojedinačno, no međutim zbog brzine smo unosili : `(broj čvorova u redku,[koordinate prvog čvora u retku], [korak za koliko se pomiču čvorovi])`

```
nds = make_nodes (5, [0., 0., 0.], [2., 0., 0.])
nds.extend (make_nodes (5, [0., 2., 0.], [2., 0., 0.]))
nds.extend (make_nodes (5, [0., 4., 0.], [2., 0., 0.]))
nds.extend (make_nodes (5, [0., 6., 0.], [2., 0., 0.]))
nds.extend (make_nodes (5, [0., 8., 0.], [2., 0., 0.]))
nds[24] = [8., 8., 4.]
nds
```

```
[[0.000000000000000, 0.000000000000000, 0.000000000000000],
[2.00000000000000, 0.000000000000000, 0.000000000000000],
[4.00000000000000, 0.000000000000000, 0.000000000000000],
[6.00000000000000, 0.000000000000000, 0.000000000000000],
[8.00000000000000, 0.000000000000000, 0.000000000000000],
[0.000000000000000, 2.00000000000000, 0.000000000000000],
[2.00000000000000, 2.00000000000000, 0.000000000000000],
[4.00000000000000, 2.00000000000000, 0.000000000000000],
[6.00000000000000, 2.00000000000000, 0.000000000000000],
[8.00000000000000, 2.00000000000000, 0.000000000000000],
[0.000000000000000, 4.00000000000000, 0.000000000000000],
[2.00000000000000, 4.00000000000000, 0.000000000000000],
[4.00000000000000, 4.00000000000000, 0.000000000000000],
[6.00000000000000, 4.00000000000000, 0.000000000000000],
[8.00000000000000, 4.00000000000000, 0.000000000000000],
[0.000000000000000, 6.00000000000000, 0.000000000000000],
[2.00000000000000, 6.00000000000000, 0.000000000000000],
[4.00000000000000, 6.00000000000000, 0.000000000000000],
[6.00000000000000, 6.00000000000000, 0.000000000000000],
[8.00000000000000, 6.00000000000000, 0.000000000000000],
[0.000000000000000, 8.00000000000000, 0.000000000000000],
[2.00000000000000, 8.00000000000000, 0.000000000000000],
[4.00000000000000, 8.00000000000000, 0.000000000000000],
[6.00000000000000, 8.00000000000000, 0.000000000000000],
[8.00000000000000, 8.00000000000000, 4.00000000000000]]
```

Zatim unosimo kabele kao: `(broj prvog čvora kabela,broj čvorova na kabelu, vrijednost koraka između dvaju susjednih čvorova na kabelu])`

```
cbls = [cable (15, 5, 1)]
cbls.append (cable (10, 5, 1))
cbls.append (cable (5, 5, 1))
cbls.append (cable (1, 5, 5))
```

```
cbls.append (cable (2, 5, 5))
cbls.append (cable (3, 5, 5))
# rubni kabeli:
cbls.append (cable (0, 5, 1))
cbls.append (cable (20, 5, 1))
cbls.append (cable (0, 5, 5))
cbls.append (cable (4, 5, 5))
cbls
```

```
[[15, 16, 17, 18, 19], [10, 11, 12, 13, 14], [5, 6, 7, 8, 9], [1, 6, 11,16,
21], [2, 7, 12, 17, 22], [3, 8, 13, 18, 23], [0, 1, 2, 3, 4], [20,21, 22,
23, 24], [0, 5, 10, 15, 20], [4, 9, 14, 19, 24]]
```

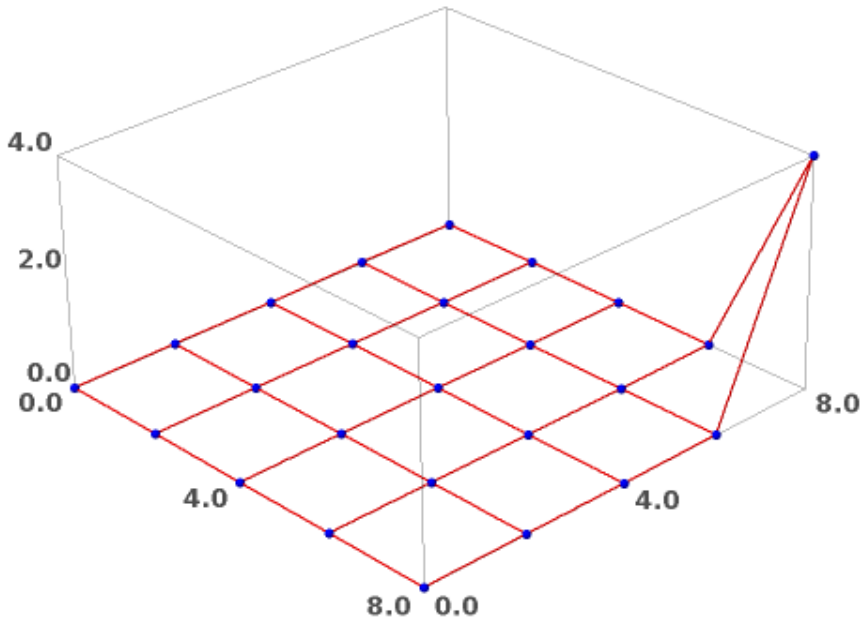
Sada kabele dijelimo na štapove koji su jednaki elementima između dvaju susje-dnih čvorova na kabelu:

```
els = make_elements_on_cables (cbls)
els
```

```
[[15, 16], [16, 17], [17, 18], [18, 19], [10, 11], [11, 12], [12, 13],[13,
14], [5, 6], [6, 7], [7, 8], [8, 9], [1, 6], [6, 11], [11, 16],[16, 21],
[2, 7], [7, 12], [12, 17], [17, 22], [3, 8], [8, 13],[13,18], [18, 23], [0,
1], [1, 2], [2, 3], [3, 4], [20, 21], [21, 22], [22,23], [23, 24], [0, 5],
[5, 10], [10, 15], [15, 20], [4, 9], [9, 14],[14, 19], [19, 24]]
```

Početna mreža kabela izgleda ovako:

```
plot3d_mesh(nds,els)
```



Zadajemo ležajne čvorove:

```
supps = [0, 4, 20, 24]
```

Štapovi na kabelima:

```
tcei = table_of_cable_element_incidences (cbls)
print_indexed (tcei)
```

```
0 :  [0, 1, 2, 3]
1 :  [4, 5, 6, 7]
2 :  [8, 9, 10, 11]
3 :  [12, 13, 14, 15]
4 :  [16, 17, 18, 19]
5 :  [20, 21, 22, 23]
6 :  [24, 25, 26, 27]
7 :  [28, 29, 30, 31]
8 :  [32, 33, 34, 35]
9 :  [36, 37, 38, 39]
```

Rubni kabeli:

```
bnd_cbls = [6, 7, 8, 9]
```

Unutarnji kabeli:

```
int_cbls = other_cables (bnd_cbls, len (cbls))
int_cbls
```

```
[0, 1, 2, 3, 4, 5]
```

Štapovi na rubnim kabelima:

```
beis = select_elements_on_cables (bnd_cbls, tcei)
beis
```

```
 [24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39]
```

Gustoće sila:

- u unutarnjim kabelima:

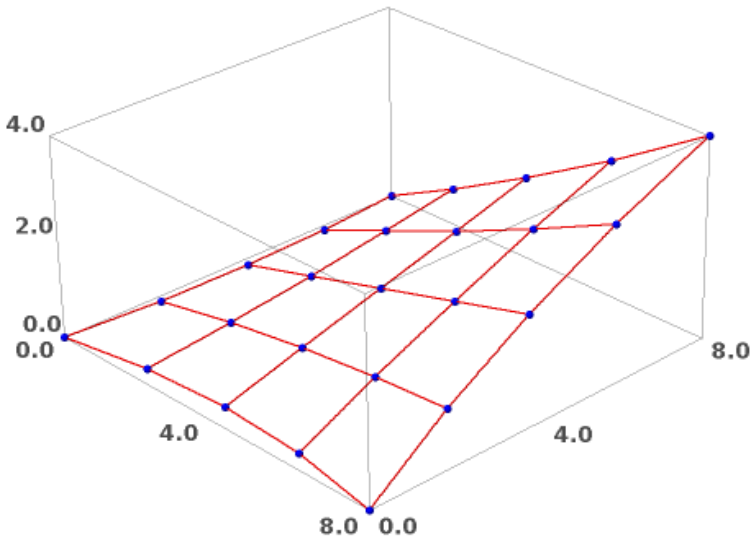
```
qs = make_force_densities (len (els))
```

- u rubnim kabelima:

```
qs = set_value_on_cables (10., bnd_cbls, tcei, qs)
```

Proračun metodom gustoće sila:

```
nc = FDM (nds, els, supps, qs)

plt1 = plot3d_mesh (nc.rows(), els)
plt1
```



Duljine štapova (iz poznatih koordinata čvorova) i sile u štapovima (f = q*l):

```
l = list_of_element_lengths (els, nc)

f = list_of_element_forces (l, qs)
```

Dobili smo vrijednosi svih duljina štapova

```
l
```

```
[1.95634043389947, 2.02330385189121, 2.03000088468678, 1.97421237815069,
1.83629254267640, 1.91427208900924, 1.91427208900924, 1.83629254267640,
1.85097479428020, 1.91036561077882, 1.90808888206519, 1.84517433881212,
1.85097479428020, 1.91036561077882, 1.90808888206519, 1.84517433881213,
1.83629254267640, 1.91427208900924, 1.91427208900924, 1.83629254267640,
1.95634043389947, 2.02330385189121, 2.03000088468678, 1.97421237815069,
2.02422151799884, 1.99726091950844, 1.99701685619016, 2.02495763206020,
2.23336348862141, 2.21761027681734, 2.23733568103318, 2.29432319969438,
2.02422151799884, 1.99726091950844, 1.99701685619016, 2.02495763206020,
2.23336348862141, 2.21761027681734, 2.23733568103319, 2.29432319969438]
```

Dobili smo vrijednosi svih sila u štapovima

```
f
```

```
[1.95634043389947, 2.02330385189121, 2.03000088468678, 1.97421237815069,
1.83629254267640, 1.91427208900924, 1.91427208900924, 1.83629254267640,
1.85097479428020, 1.91036561077882, 1.90808888206519, 1.84517433881212,
1.85097479428020, 1.91036561077882, 1.90808888206519, 1.84517433881213,
1.83629254267640, 1.91427208900924, 1.91427208900924, 1.83629254267640,
1.95634043389947, 2.02330385189121, 2.03000088468678, 1.97421237815069,
20.2422151799884, 19.9726091950844, 19.9701685619016, 20.2495763206020,
22.3336348862141, 22.1761027681734, 22.3733568103318, 22.9432319969438,
20.2422151799884, 19.9726091950844, 19.9701685619016, 20.2495763206020,
22.3336348862141, 22.1761027681734, 22.3733568103319, 22.9432319969438]
```

Sile u pojedinim štapovima se razlikuju. To je moguće ostvariti tako da se kabeli u čvorovima međusobno povežu.

Sile u štapovima rubnih kabela:

```
f_bnd = get_values_on_cables (bnd_cbls, tcei, f)
```

```
f_bnd
```

```
[10.1666054911088, 10.0793761664939, 10.0964685181053, 10.2083773003431,
9.96953958233539, 9.90753174152372, 9.99239239322426, 10.2260236747238,
10.1666054911088, 10.0793761664939, 10.0964685181053, 10.2083773003431,
9.96953958233548, 9.90753174152381, 9.99239239322435, 10.2260236747239]
```

Sile u štapovima unutarnjih kabela:

```
f_int = get_values_on_cables (int_cbls, tcei, f)
```

Zbog simetričog oblika mreže i sile u štapovima pojedinih kabela moraju biti jednake.

- sile u štapovima kabela iznose:

```
get_values_on_cables ([0], tcei, f)
```

```
[1.95634043389947, 2.02330385189121, 2.03000088468678, 1.97421237815069]
```

```
get_values_on_cables ([1], tcei, f)
```

```
[1.83629254267640, 1.91427208900924, 1.91427208900924, 1.83629254267640]
```

```
get_values_on_cables ([2], tcei, f)
```

```
[1.85097479428020, 1.91036561077882, 1.90808888206519, 1.84517433881212]
```

```
get_values_on_cables ([3], tcei, f)
```

```
[1.85097479428020, 1.91036561077882, 1.90808888206519, 1.84517433881213]
```

```
get_values_on_cables ([4], tcei, f)
```

```
[1.83629254267640, 1.91427208900924, 1.91427208900924, 1.83629254267640]
```

```
get_values_on_cables ([5], tcei, f)
```

```
[1.95634043389947, 2.02330385189121, 2.03000088468678, 1.97421237815069]
```

```
get_values_on_cables ([6], tcei, f)
```

```
[10.1666054911088, 10.0793761664939, 10.0964685181053, 10.2083773003431]
```

```
get_values_on_cables ([7], tcei, f)
```

```
[9.96953958233539, 9.90753174152372, 9.99239239322426, 10.2260236747238]
```

```
get_values_on_cables ([8], tcei, f)
```

```
[10.1666054911088, 10.0793761664939, 10.0964685181053, 10.2083773003431]
```

```
get_values_on_cables ([9], tcei, f)
```

```
[9.96953958233548, 9.90753174152381, 9.99239239322435, 10.2260236747239]
```

- najveća i najmanja sila u štapovima unutarnjih kabela:

```
max (f_int), min (f_int)
```

```
(2.03000088468678, 1.83629254267640)
```

- najveća i najmanja sila u štapovima rubnh kabela:

```
max (f_bnd), min (f_bnd)
```

```
(22.9432319969438, 19.9701685619016)
```

- duljine štapova rubnih kabela:

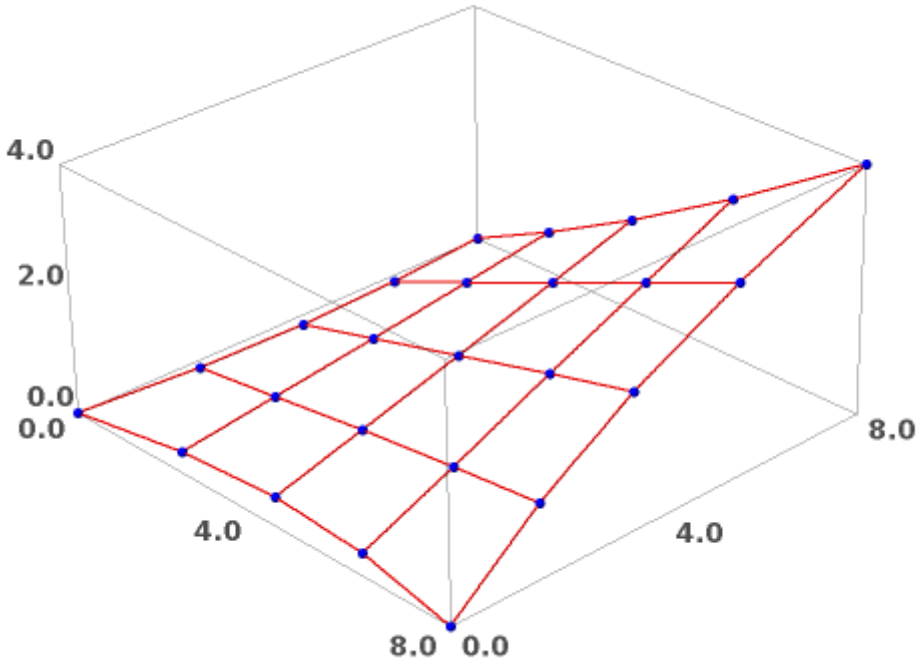
```
bels = get_values_on_cables (bnd_cbls, tcei, l)
```

```
bels
```

```
[2.02422151799884, 1.99726091950844, 1.99701685619016, 2.02495763206020,
2.23336348862141, 2.21761027681734, 2.23733568103318, 2.29432319969438,
2.02422151799884, 1.99726091950844, 1.99701685619016, 2.02495763206020,
2.23336348862141, 2.21761027681734, 2.23733568103319, 2.29432319969438]
```

Ponavljanje proračuna metodom gustoća sila 5 puta (s izjednačavanjem sila u štapovima):
- pomoću funkcije multiStepFDM pokreće se iterativni postupak određivanja ravnotežnog položaja.

```
nc, f = multistepFDM2 (nds, els, supps, qs, 5)

plot3d_mesh (nc.rows(), els)
```



- duljine štapova:

```
l1 = list_of_element_lengths (els, nc.rows())
```

```
l1
```

```
[1.83005559955882, 2.02090952468133, 2.06888836511126, 1.93455151680935,
1.70171167609665, 1.95415426807835, 1.97672876896680, itd.]
```

- na rubnim kabelima:

```
bels = get_values_on_cables (bnd_cbls, tcei, l1)
bels
```

```
[2.06718629801691, 1.94707168106293, 1.95245243898225, 2.08327360622697,
2.19245863806933, 2.10554914839902, 2.20074519246812, 2.50245107041511,
2.06718629801691, 1.94707168106293, 1.95245243898226, 2.08327360622697,
2.19245863806933, 2.10554914839902, 2.20074519246812, 2.50245107041511]
```

U nastavku više ne mijenjamo duljine štapova na rubnim kabelima.
- parovi (indeks štapa, duljina štapa) za zadavanje duljina, pridruženi pomoću funkcije `zip`:

```
beils = zip (beis, bels)
beils
```

```
[(24, 2.06718629801691), (25, 1.94707168106293), (26, 1.95245243898225),
(27, 2.08327360622697), (28, 2.19245863806933), (29, 2.10554914839902),
(30, 2.20074519246812), (31, 2.50245107041511), (32, 2.06718629801691),
(33, 1.94707168106293), (34, 1.95245243898226), (35, 2.08327360622697),
(36, 2.19245863806933), (37, 2.10554914839902), (38, 2.20074519246812),
(39, 2.50245107041511)]
```

- parovi (indeks štapa, sila u štapu) za zadavanje sila:
 -- indeksi unutarnjih štapova:

```
ieis = select_elements_on_cables (int_cbls, tcei)
ieis
```

```
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19,
20, 21, 22, 23]
```

-- sile u štapovima (u svim unutrašnjim štapovima zadajemo sile jednake 1):

```
iefs = [1. for i in ieis]
```

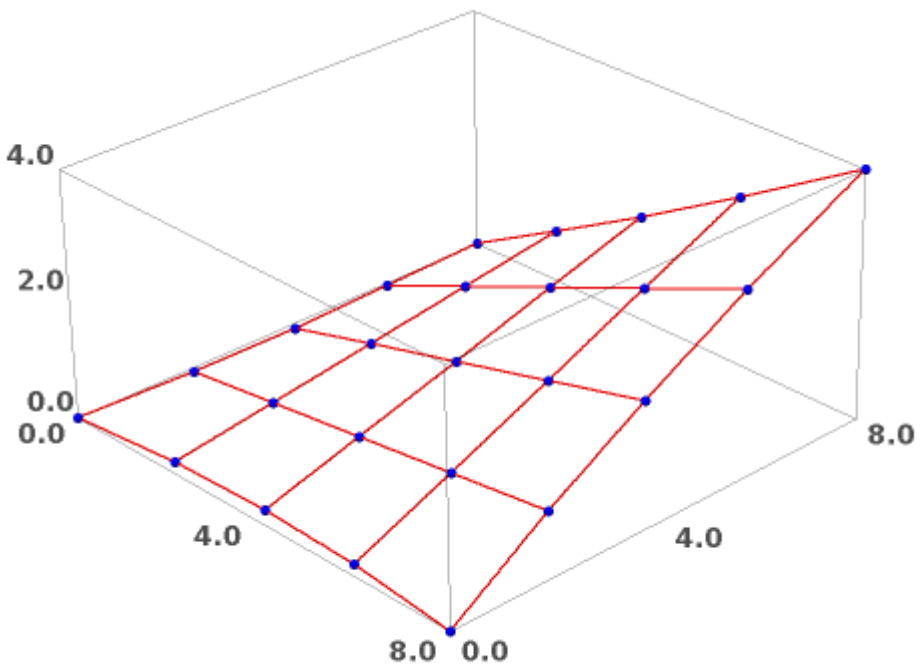
-- parovi:

```
ieifs = zip (ieis, iefs)
ieifs
```

```
[(0, 1.00000000000000), (1, 1.00000000000000), (2, 1.00000000000000),
(3, 1.00000000000000), (4, 1.00000000000000), itd.]
```

Ponavljanje proračuna 10 puta:

```
nc, f = multistepFDM (nds, els, supps, qs, ieifs, beils, 10)
plot3d_mesh (nc.rows(), els)
```




```
f
```

```
[0.999861875983436, 0.999757689048192, 0.999514765258799,1.00013920476133,
0.999831875730992, 0.999662335968796,0.999356952968110, itd.]
```

Ponavljanje proračuna do zadovoljenja zadanih točnosti, s time da iteraciju ograničimo na 1000 koraka u slučaju divergencije (proračun će se automatski zaustaviti kada rješenje najbliže konvergira zadanom rješenju)

- 1.e-4 sile, 1.e-2 duljine:

```
nc, f = multistepFDM_wtol (nds, els, supps, qs, ieifs, beils,
1.e-4, 1.e-2, 1000)
```

```
steps: 335
maximal force error:  0.0000997014426037390
maximal length error: 0.000859880427034554
```

- 1.e-4 sile, 1.e-3 duljine:

```
nc, f = multistepFDM_wtol (nds, els, supps, qs, ieifs, beils,
1.e-4, 1.e-3, 1000)
```

```
steps: 335
maximal force error:  0.0000997014426037390
maximal length error: 0.000859880427034554
```

- 1.e-4 sile, 1.e-4 duljine:

```
nc, f = multistepFDM_wtol (nds, els, supps, qs, ieifs, beils,
1.e-4, 1.e-4, 1000)
```

```
steps: 510
maximal force error:  0.0000119311677615475
maximal length error: 0.0000990635903939108
```

```
f
```

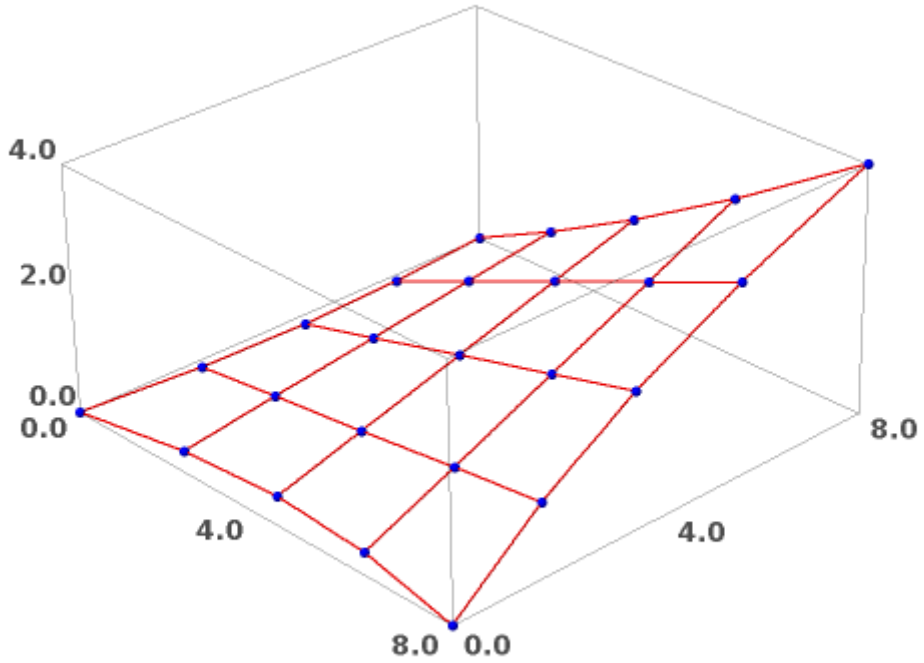
```
[0.999994213678061, 1.00000089026309, 1.00000069661154,0.999991321367080,
0.999991698729755, 1.00000074614789,1.00000063078330, itd.]
```

```
f_bnd = get_values_on_cables (bnd_cbls, tcei, f)
```

```
max (f_bnd), min (f_bnd)
```

```
(10.2260236747239, 9.90753174152372)
```

```
plot3d_mesh (nc.rows(), els)
```



Zadane duljine štapova rubnih kabela - srednje vrijednosti duljina nakon 5 koraka FDM-a:

```
bels
```

```
[2.06718629801691, 1.94707168106293, 1.95245243898225, 2.08327360622697,
2.19245863806933, 2.10554914839902, 2.20074519246812, 2.50245107041511,
2.06718629801691, 1.94707168106293, 1.95245243898226, 2.08327360622697,
2.19245863806933, 2.10554914839902, 2.20074519246812, 2.50245107041511]
```

```
le1 = mean (bels[0:4])
le2 = mean (bels[4:8])
le3 = mean (bels[8:12])
le4 = mean (bels[12:16])
le1, le2, le3, le4
```

```
2.01249600607227, 2.25030101233789, 2.01249600607227, 2.25030101233789)
```

```
bels2 = [le1, le1, le1, le1, le2, le2, le2, le2, le3, le3,
le3, le3, le4, le4, le4, le4]

beils2 = zip (beis, bels2)
beils2
```

```
[(24, 2.01249600607227), (25, 2.01249600607227), (26, 2.01249600607227),
(27, 2.01249600607227), (28, 2.25030101233789), (29, 2.25030101233789),
(30, 2.25030101233789), (31, 2.25030101233789), (32, 2.01249600607227),
(33, 2.01249600607227), (34, 2.01249600607227), (35, 2.01249600607227),
(36, 2.25030101233789), (37, 2.25030101233789), (38, 2.25030101233789),
(39, 2.25030101233789)]
```

```
time nc2, f2 = multistepFDM_wtol (nds, els, supps, qs, ieifs,
beils2, 1.e-4, 1.e-3, 1000)
```

```
steps: 347
maximal force error:  0.0000994867607124750
maximal length error: 0.000730311685007035
Time: CPU 5.68 s, Wall: 5.70 s
```

```
f2
```

```
[0.999959893527211, 1.00000749306476, 1.00000446966676,0.999929508685902,
0.999939576170726, 1.00000583324877,1.00000448585173, itd.]
```

U čvoru 15 vektorski zbrajamo sile iz štapova. Njihova suma mora biti 0 da bi smo imali ravnotežu:

```
v0 = vector (nc2[16] - nc2[15])
v35 = vector (nc2[20] - nc2[15])
v34 = vector (nc2[10] - nc2[15])
v0, v35, v34
```

```
((1.72854035439594, 0.00848895554210394, 0.663079685154796),
(-0.281673521092412, 1.99046944583173, -0.0827989525489769),
(0.0989905961421713, -2.00952098360947, 0.0124903931491443))
```

```
f2[0] * v0/norm (v0) + f2[35] * v35/norm (v35) + f2[34] *
v34/norm (v34)
```

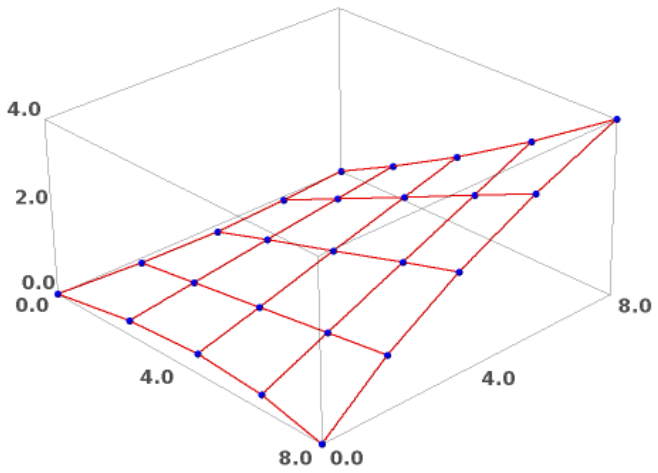
```
(0.000000000000000, -5.32907051820075e-15, 1.11022302462516e-16)
```

Dokazali smo da su u čvoru 15 sve sile u ravnoteži.

```
f_bnd = get_values_on_cables (bnd_cbls, tcei, f2)
max (f_bnd), min (f_bnd)
```

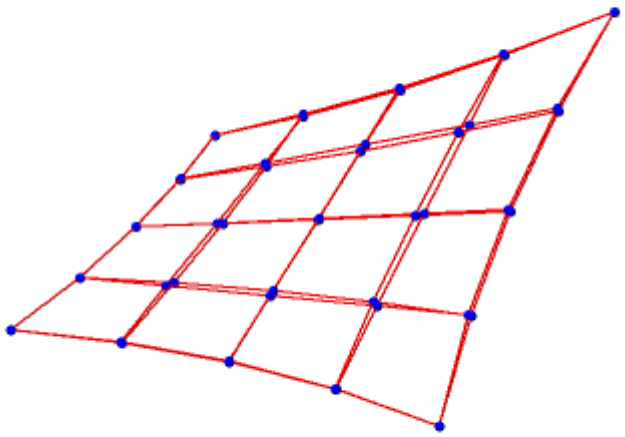
```
(10.3090382854024, 9.93219208134452)
```

```
plt2 = plot3d_mesh (nc2.rows(), els)
plt2
```



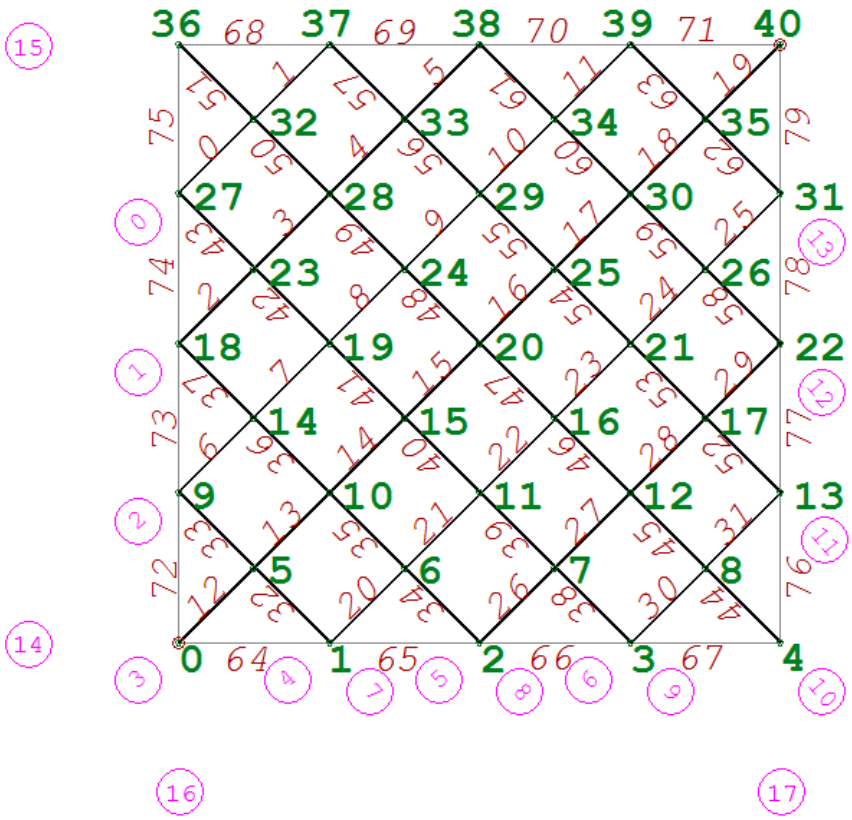
Usporedba mreže oblikovane metodom gustoće sila i minimalne mreže:

```
(plt1 + plt2).rotateZ (-pi/8).show (frame = False)
```



Primjer 2.

Tražimo stabilni oblik mreže kabela koji se nalaži u ravnoteži. Mreža se sastoji od 18 *[0-17]* kabela koji se sastoje od 80 *[0-79]* štapova i ukupno 41 *[0-40]* čvorova. U mreži imamo 4 *[14-17]* rubna kabela na kojima se nalazi 16 *[64-79]* rubnih štapova. Ostalih 64 *[0-63]* štapova nalazi se na unutarnih 14 *[0-13]* kabela. Konstrukcija ima 4 ležajna čvora, od kojih 2 *[4, 36]* leže u donjoj ravnini, a druga dva *[0, 40]* u go-rnjoj ravnini.



Najprije zadamo čvorove mreže:

- ovdje smo mogli svaku točku mreže upisati pojedinačno, no međutim zbog brzine smo unosili : `(broj čvorova u redku,[koordinatu prvog čvora u retku], [korak za koliko se pomiću čvorovi])`

```
nds = make_nodes (5, [0., 0., 0.], [2., 0., 0.])
nds.extend (make_nodes (4, [1., 1., 0.], [2., 0., 0.]))
nds.extend (make_nodes (5, [0., 2., 0.], [2., 0., 0.]))
nds.extend (make_nodes (4, [1., 3., 0.], [2., 0., 0.]))
nds.extend (make_nodes (5, [0., 4., 0.], [2., 0., 0.]))
```

```
nds.extend (make_nodes (4, [1., 5., 0.], [2., 0., 0.]))
nds.extend (make_nodes (5, [0., 6., 0.], [2., 0., 0.]))
nds.extend (make_nodes (4, [1., 7., 0.], [2., 0., 0.]))
nds.extend (make_nodes (5, [0., 8., 0.], [2., 0., 0.]))
nds[0] = [0., 0., 4.]
nds[40] = [8., 8., 4.]
```

Zatim unosimo kabele kao: `(broj prvog čvora kabela,broj čvorova na kabelu, vrijednost koraka između dvaju susjednih čvorova na kabelu])`

```
cbls = [cable (27, 3, 5)]
cbls.append (cable (18, 5, 5))
cbls.append (cable (9, 7, 5))
cbls.append (cable (0, 9, 5))
cbls.append (cable (1, 7, 5))
cbls.append (cable (2, 5, 5))
cbls.append (cable (3, 3, 5))
cbls.append (cable (1, 3, 4))
cbls.append (cable (2, 5, 4))
cbls.append (cable (3, 7, 4))
cbls.append (cable (4, 9, 4))
cbls.append (cable (13, 7, 4))
cbls.append (cable (22, 5, 4))
cbls.append (cable (31, 3, 4))
 # rubni kabeli:
cbls.append (cable (0, 5, 1))
cbls.append (cable (36, 5, 1))
cbls.append (cable (0, 5, 9))
cbls.append (cable (4, 5, 9))
cbls
```

```
[[27, 32, 37], [18, 23, 28, 33, 38], [9, 14, 19, 24, 29, 34, 39], itd.]
```

Sada kabele podijelimo na štapove koji su jednaki elementima između dvaju susjednih čvorova na kabelu:

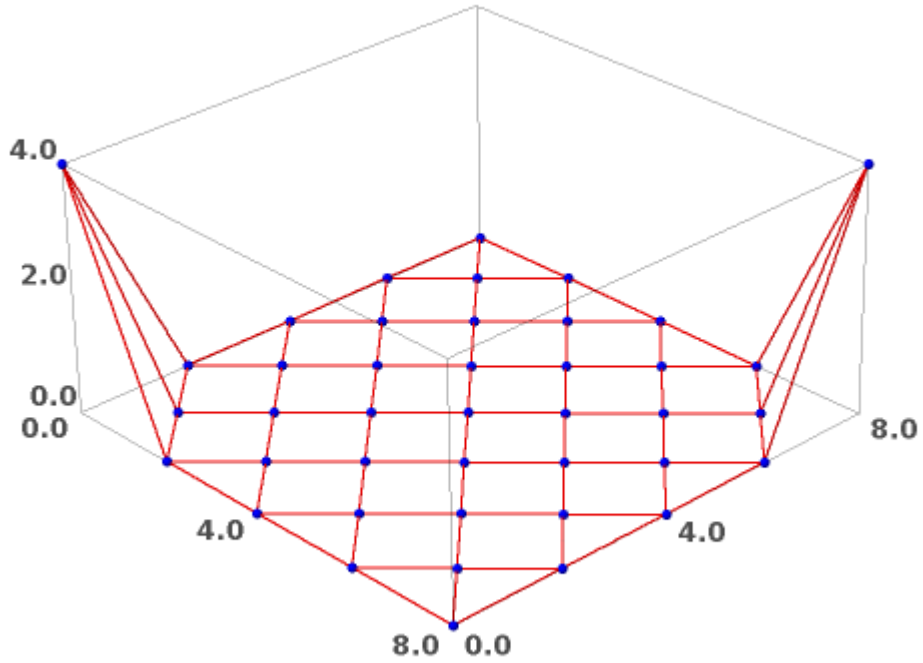
```
els = make_elements_on_cables (cbls)
```

```
els
```

```
[[27, 32, 37], [18, 23, 28, 33, 38], [9, 14, 19, 24, 29, 34, 39], itd.]
```

Početna mreža kabela izgleda ovako

```
plot3d_mesh (nds, els)
```



Zadajemo ležajne čvorove:

```
supps = [0, 4, 36, 40]
```

Štapovi na kabelima:

```
tcei = table_of_cable_element_incidences (cbls)
print_indexed (tcei)
```

```
0 :  [0, 1]
Itd.
```

Rubni kabeli:

```
bnd_cbls = [14, 15, 16, 17]
```

Unutarnji kabeli:

```
int_cbls = other_cables (bnd_cbls, len (cbls))
int_cbls
```

```
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13]
```

Štapovi na rubnim kabelima:

```
beis = select_elements_on_cables (bnd_cbls, tcei)
beis
```

```
 [64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79]
```

Gustoće sila:

- u unutarnjim kabelima:

```
qs = make_force_densities (len (els))
```

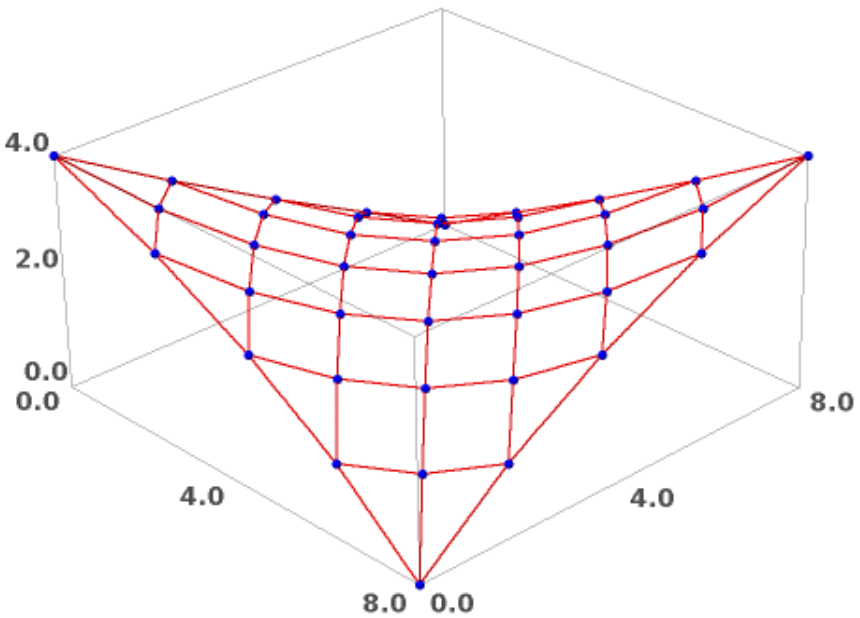
- u rubnim kabelima:

```
qs = set_value_on_cables (10., bnd_cbls, tcei, qs)
```

Proračun metodom gustoće sila:

```
nc = FDM (nds, els, supps, qs)
```

```
plt1 = plot3d_mesh (nc.rows(), els)
plt1
```



Duljine štapova (iz poznatih koordinata čvorova) i sile u štapovima (f = q l):

```
l = list_of_element_lengths (els, nc)
f = list_of_element_forces (l, qs)
```

Dobili smo vrijednosi svih duljina štapova:

```
l
```

```
[1.24041349274239, 1.24041349274239, 1.35142523669045, itd.]
```

Dobili smo vrijednosi svih sila u štapovima

```
f
```

```
[1.24041349274239, 1.24041349274239, 1.35142523669045, itd.]
```

Sile u štapovima rubnih kabela:

```
f_bnd = get_values_on_cables (bnd_cbls, tcei, f)
```

```
f_bnd
```

```
[22.6688132306021, 22.2246518145741, 22.2246518145741, itd.]
```

Sile u štapovima unutarnjih kabela:

```
f_int = get_values_on_cables (int_cbls, tcei, f)
```

Najveća i najmanja sila u štapovima unutarnjih kabela:

```
max (f_int), min (f_int)
```

```
(1.78101405323454, 1.24041349274239)
```

Najveća i najmanja sila u štapovima rubnh kabela:

```
max (f_bnd), min (f_bnd)
```

```
(22.6688132306021, 22.2246518145740)
```

Duljine štapova rubnih kabela:

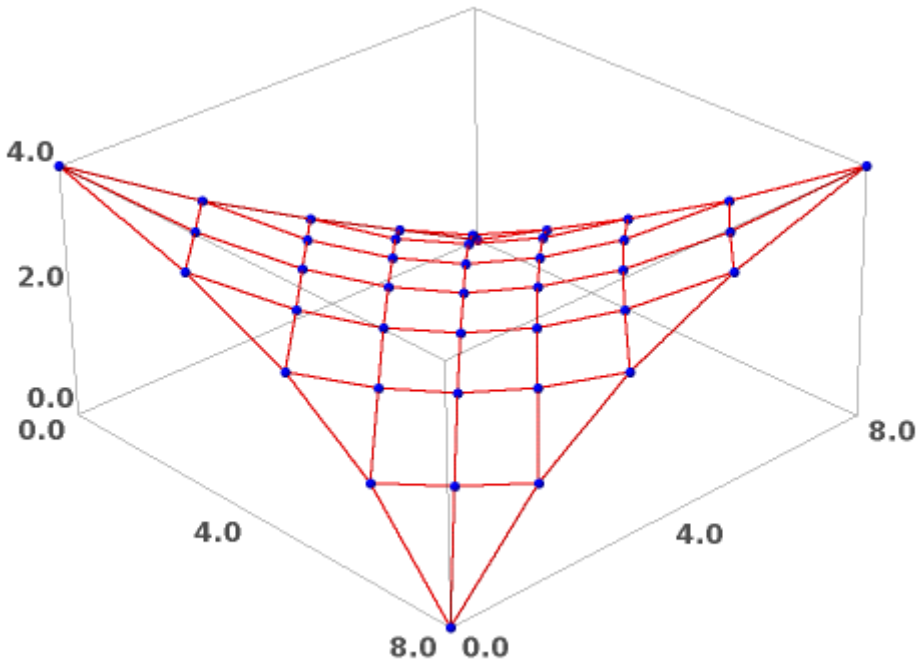
```
bels = get_values_on_cables (bnd_cbls, tcei, l)
```

```
bels
```

```
[2.26688132306021, 2.22246518145741, 2.22246518145741, itd.]
```

Ponavljanje proračuna metodom gustoća sila 5 puta (s izjednacavanjem sila u štapovima):
- pomoću funkcije multiStepFDM pokreće se iterativni postupak određivanja ravnotežnog položaja.

```
nc, f = multistepFDM2 (nds, els, supps, qs, 5)

plot3d_mesh (nc.rows(), els)
```



- duljine štapova:

```
l1 = list_of_element_lengths (els, nc.rows())
```

- na rubnim kabelima:

```
bels = get_values_on_cables (bnd_cbls, tcei, l1)
```

```
bels
```

```
[2.43120843422434, 2.10122363830448, 2.10122363830448, itd.]
```

- parovi (indeks štapa, duljina štapa) za zadavanje duljina:

```
beils = zip (beis, bels)
beils
```

```
[(64, 2.43120843422434), (65, 2.10122363830448), itd.]
```

- parovi (indeks štapa, sila u štapu) za zadavanje sila:
-- indeksi unutarnjih štapova:

```
ieis = select_elements_on_cables (int_cbls, tcei)

ieis
```

```
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, itd.]
```

-- sile u štapovima:

```
iefs = [1. for i in ieis]
```

-- parovi:

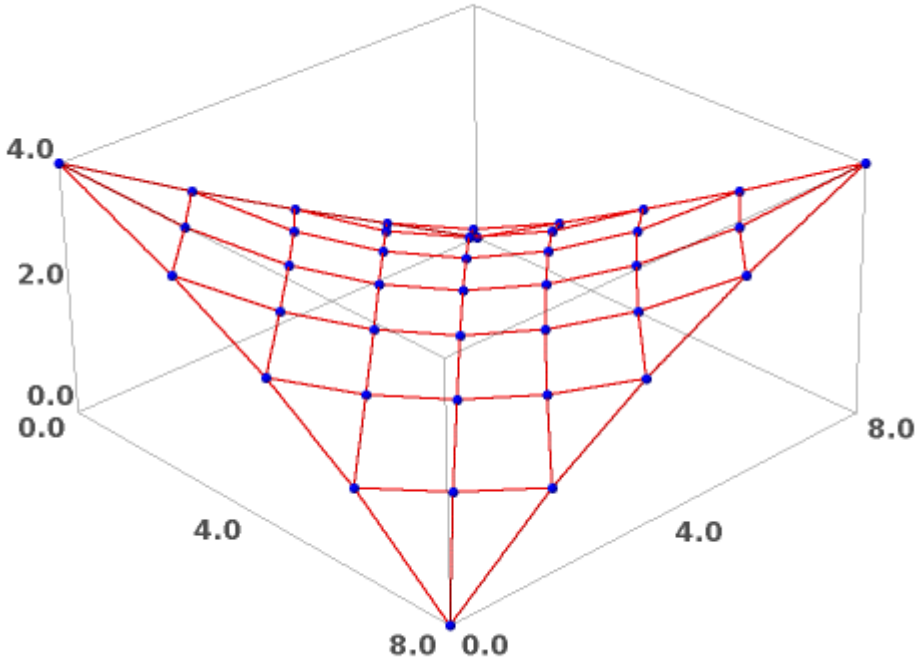
```
ieifs = zip (ieis, iefs)

ieifs
```

```
[(0, 1.00000000000000), (1, 1.00000000000000), (2, 1.00000000000000),itd.]
```

Ponavljanje proračuna 10 puta:

```
nc, f = multistepFDM (nds, els, supps, qs, ieifs, beils, 10)
```

```
plot3d_mesh (nc.rows(), els)
```




```
f
```

```
[0.998530306185056, 0.998530306185053, 1.00062690109457, itd.]
```

Ponavljanje proračuna do zadovoljenja zadanih točnosti, s time da iteraciju ograničimo na 1000 koraka u slučaju divergencije (proračun će se automatski zaustaviti kada riješenje najbliže konvergira zadanom rješenju)

- 1.e-4 sile, 1.e-2 duljine:

```
nc, f = multistepFDM_wtol (nds, els, supps, qs, ieifs, beils,
1.e-4, 1.e-2, 1000)
```

```
steps: 206
maximal force error:  0.0000999505088485897
maximal length error: 0.000506653466498364
```

- 1.e-4 sile, 1.e-3 duljine:

```
nc, f = multistepFDM_wtol (nds, els, supps, qs, ieifs, beils,
1.e-4, 1.e-3, 1000)
```

```
steps: 206
maximal force error:  0.0000999505088485897
maximal length error: 0.000506653466498364
```

- 1.e-4 sile, 1.e-4 duljine:

```
nc, f = multistepFDM_wtol (nds, els, supps, qs, ieifs, beils,
1.e-4, 1.e-4, 1000)
```

```
steps: 268
maximal force error:  0.0000197780647058776
maximal length error: 0.0000981179482923977
```

```
f
```

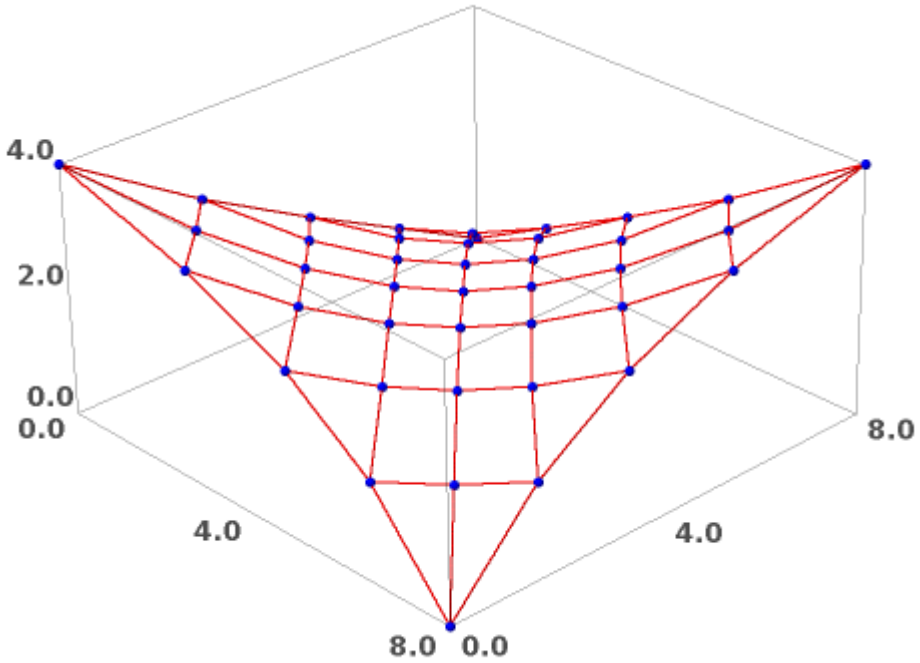
```
[0.999985198891379, 0.999985198891380, 0.999998983378320,0.999982010816570,
0.999982010816570, 0.999998983378320,1.00000356581564, itd.]
```

```
f_bnd = get_values_on_cables (bnd_cbls, tcei, f)
```

```
max (f_bnd), min (f_bnd)
```

```
(10.3076375137869, 10.0311013187105)
```

```
plot3d_mesh (nc.rows(), els)
```



Zadane duljine štapova rubnih kabela - srednje vrijednosti duljina nakon 5 koraka FDM-a:

```
bels
```

```
[2.43120843422434, 2.10122363830448, 2.10122363830448, 2.43120843422434,
2.43120843422434, 2.10122363830448, 2.10122363830449, itd.]
```

```
le1 = mean (bels[0:4])
le2 = mean (bels[4:8])
le3 = mean (bels[8:12])
le4 = mean (bels[12:16])
le1, le2, le3, le4
```

```
(2.26621603626441, 2.26621603626441, 2.26621603626441, 2.26621603626441)
```

```
bels2 = [le1, le1, le1, le1, le2, le2, le2, le2, le3, le3,
le3, le3, le4, le4, le4, le4]
```

```
beils2 = zip (beis, bels2)

beils2
```

```
[(64, 2.26621603626441), (65, 2.26621603626441), (66, 2.26621603626441),
(67, 2.26621603626441), (68, 2.26621603626441), itd.]
```

```
time nc2, f2 = multistepFDM_wtol (nds, els, supps, qs, ieifs,
beils2, 1.e-4, 1.e-3, 1000)
```

```
steps: 213
maximal force error:  0.0000994287810471839
maximal length error: 0.000440459126255455
Time: CPU 8.43 s, Wall: 8.44 s
```

```
f2
```

```
[0.999929146683196, 0.999929146683195, 0.999995103953985,0.999910883494553,
0.999910883494557, 0.999995103953984,1.00001663238127, 0.999998938851058,
0.999901530245995,0.999901530245991, 0.999998938851058, itd.]
```

```
v0 = vector (nc2[32] - nc2[27])
v43 = vector (nc2[23] - nc2[27])
v74 = vector (nc2[18] - nc2[27])
v75 = vector (nc2[36] - nc2[27])

v0, v43, v74, v75
```

```
((0.758904112280723, 0.719851788194854, -0.0588589791392199),
(1.24760766283199, -1.10113501038904, 0.662357943210379),
(0.170330487651542, -2.03651957342251, 0.978424508073645),
(-0.484724526101919, 1.96348042657749, -1.02157549192636))
```

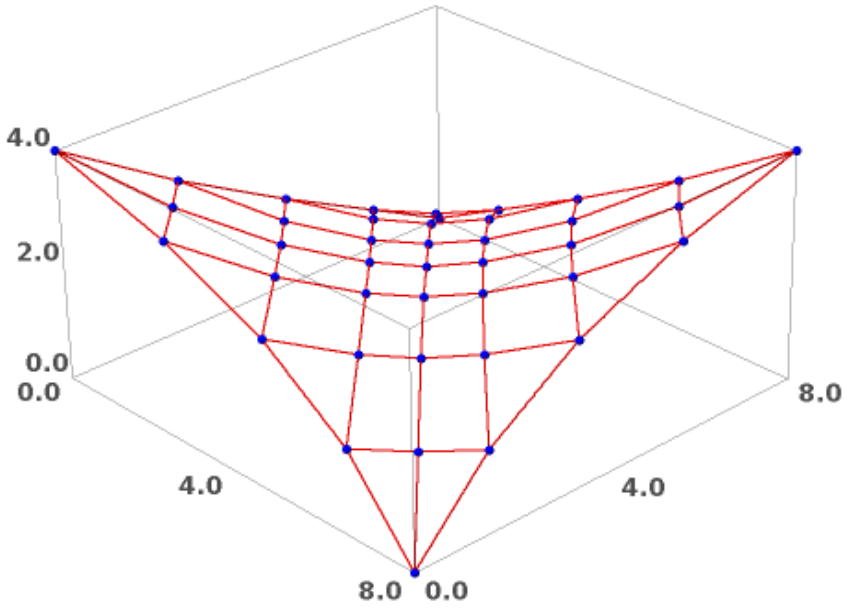
```
f2[0] * v0/norm (v0) + f2[43] * v43/norm (v43) + f2[74] *
v74/norm (v74) + f2[75] * v75/norm (v75)
```

```
(8.88178419700125e-16, 1.77635683940025e-14, 0.000000000000000)
```

```
f_bnd = get_values_on_cables (bnd_cbls, tcei, f2)

max (f_bnd), min (f_bnd)
```

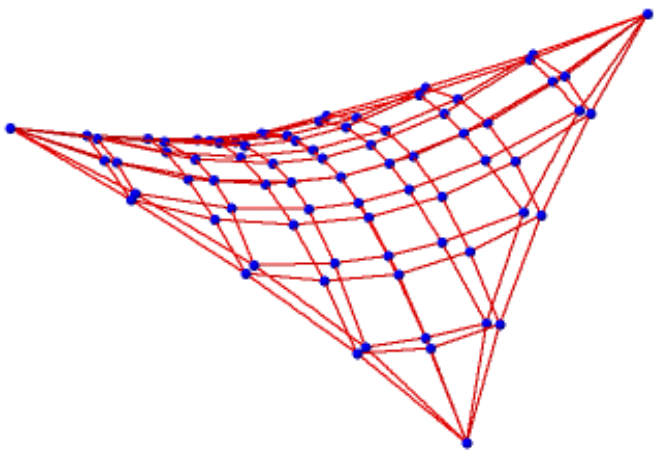
```
(10.0880064212862, 9.80648618554125)
```

```
plt2 = plot3d_mesh (nc2.rows(), els)

plt2
```



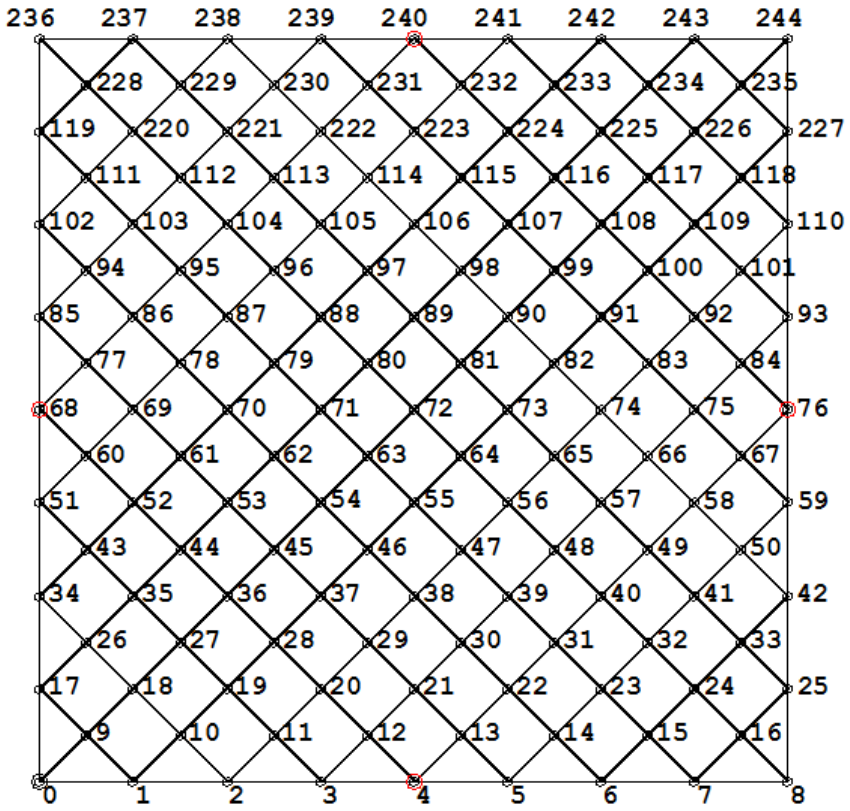
Usporedba mreže oblikovane metodom gustoće sila i minimalne mreže

```
(plt1 + plt2).rotateZ (-pi/8).show (frame = False)
```



Primjer 3.

Tražimo stabilni oblik mreže kabela koji se nalaži u ravnoteži. Mreža se sastoji od 34 *[0-33]* kabela koji se sastoje od 288 *[0-287]* štapova i ukupno 145 *[0-144]* čvorova. U mreži imamo 4 *[30-33]* rubna kabela na kojima se nalazi 32 *[256-287]* rubnih štapova. Ostalih 256 *[0-255]* štapova nalazi se na unutarnih 30 *[0-29]* kabela. Konstrukcija ima 8 ležajnih čvorova, od kojih 4 *[0, 8, 136, 144]* leže u donjoj ravnini, a drugih 4 *[4, 68, 76, 14]* u gornjoj ravnini.



Čvorovi:

```
nds = make_nodes (9, [0., 0., 0.], [1., 0., 0.])
nds.extend (make_nodes (8, [0.5, 0.5, 0.], [1., 0., 0.]))
nds.extend (make_nodes (9, [0., 1., 0.], [1., 0., 0.]))
nds.extend (make_nodes (8, [0.5, 1.5, 0.], [1., 0., 0.]))
nds.extend (make_nodes (9, [0., 2., 0.], [1., 0., 0.]))
nds.extend (make_nodes (8, [0.5, 2.5, 0.], [1., 0., 0.]))
nds.extend (make_nodes (9, [0., 3., 0.], [1., 0., 0.]))
nds.extend (make_nodes (8, [0.5,3.5, 0.], [1., 0., 0.]))
nds.extend (make_nodes (9, [0., 4., 0.], [1., 0., 0.]))
nds.extend (make_nodes (8, [0.5,4.5, 0.], [1., 0., 0.]))
nds.extend (make_nodes (9, [0., 5., 0.], [1., 0., 0.]))
nds.extend (make_nodes (8, [0.5,5.5, 0.], [1., 0., 0.]))
```

```
nds.extend (make_nodes (9, [0., 6., 0.], [1., 0., 0.]))
nds.extend (make_nodes (8, [0.5,6.5, 0.], [1., 0., 0.]))
nds.extend (make_nodes (9, [0., 7., 0.], [1., 0., 0.]))
nds.extend (make_nodes (8, [0.5,7.5, 0.], [1., 0., 0.]))
nds.extend (make_nodes (9, [0., 8., 0.], [1., 0., 0.]))
nds[0] = [0., 0., 0.]
nds[4] = [4., 0., 4.]
nds[8] = [8., 0., 0.]
nds[68] = [0., 4., 4.]
nds[76] = [8., 4., 4.]
nds[136] = [0., 8., 0.]
nds[140] = [4., 8., 4.]
nds[144] = [8., 8., 0.]
```

Kabeli:

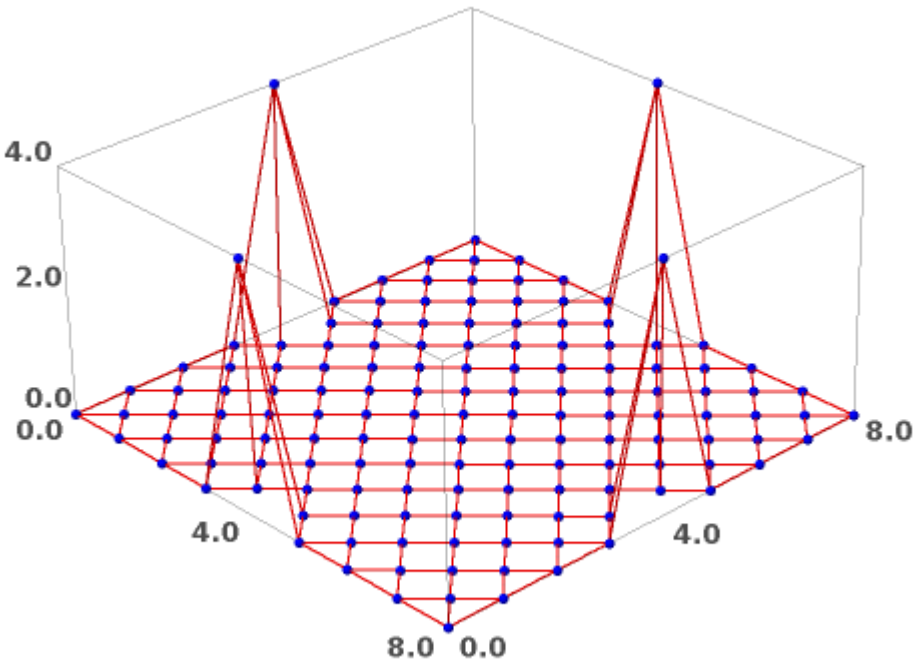
```
cbls = [cable (119, 3, 9)]
cbls.append (cable (102, 5, 9))
cbls.append (cable (85, 7, 9))
cbls.append (cable (68, 9, 9))
cbls.append (cable (51, 11, 9))
cbls.append (cable (34, 13, 9))
cbls.append (cable (17, 15, 9))
cbls.append (cable (0, 17, 9))
cbls.append (cable (1, 15, 9))
cbls.append (cable (2, 13, 9))
cbls.append (cable (3, 11, 9))
cbls.append (cable (4, 9, 9))
cbls.append (cable (5, 7, 9))
cbls.append (cable (6, 5, 9))
cbls.append (cable (7, 3, 9))
cbls.append (cable (1, 3, 8))
cbls.append (cable (2, 5, 8))
cbls.append (cable (3, 7, 8))
cbls.append (cable (4, 9, 8))
cbls.append (cable (5, 11, 8))
cbls.append (cable (6, 13, 8))
cbls.append (cable (7, 15, 8))
cbls.append (cable (8, 17, 8))
cbls.append (cable (25, 15, 8))
cbls.append (cable (42, 13, 8))
cbls.append (cable (59, 11, 8))
cbls.append (cable (76, 9, 8))
cbls.append (cable (93, 7, 8))
cbls.append (cable (110, 5, 8))
cbls.append (cable (127, 3, 8))
# rubni kabeli:
cbls.append (cable (0, 9, 1))
cbls.append (cable (136, 9, 1))
cbls.append (cable (0, 9, 17))
cbls.append (cable (8, 9, 17))
```

Elementi (štapovi):

```
els = make_elements_on_cables (cbls)
```

Ispis početne mreže:

```
plot3d_mesh (nds, els)
```



Ležajevi:

```
supps = [0, 4, 8, 68, 76, 136, 140,  144]
```

Štapovi na kabelima:

```
tcei = table_of_cable_element_incidences (cbls)
print_indexed (tcei)
```

```
0 :  [0, 1]
1 :  [2, 3, 4, 5]
2 :  [6, 7, 8, 9, 10, 11]
3 :  [12, 13, 14, 15, 16, 17, 18, 19]
4 :  [20, 21, 22, 23, 24, 25, 26, 27, 28, 29]
5 :  [30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41]
6 :  [42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55]
7 :  [56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71]
8 :  [72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85]
9 :  [86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97]
Itd.
```

Rubni i unutarnji kabeli:

```
bnd_cbls = [30, 31, 32, 33]

int_cbls = other_cables (bnd_cbls, len (cbls))
```

Štapovi na rubnim kabelima:

```
beis = select_elements_on_cables (bnd_cbls, tcei)
```

Gustoće sila:

- u unutarnjim kabelima:

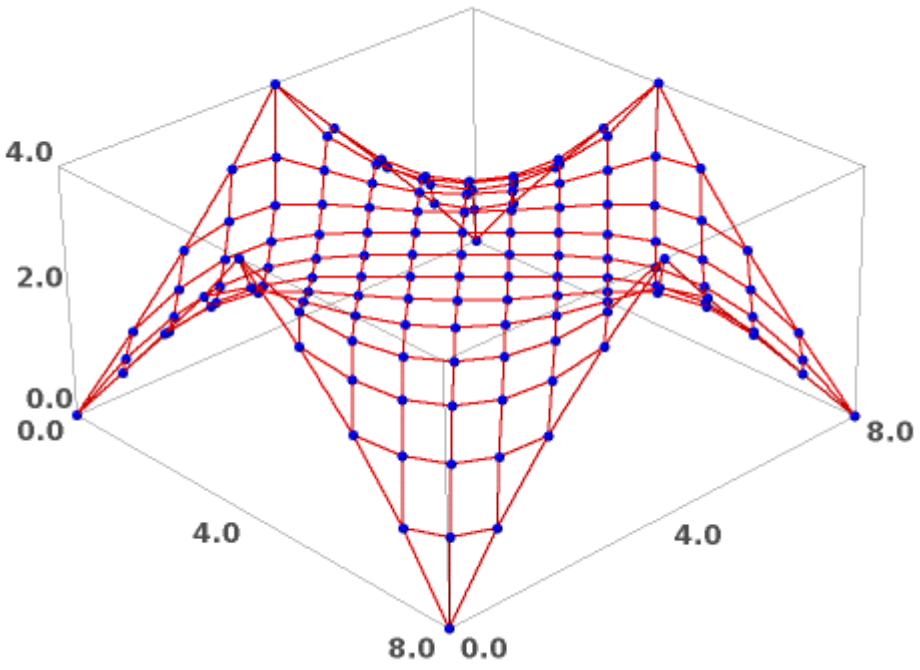
```
qs = make_force_densities (len (els))
```

- u rubnim kabelima:

```
qs = set_value_on_cables (10., bnd_cbls, tcei, qs)
```

Proračun metodom gustoće sila:

```
nc = FDM (nds, els, supps, qs)

plt1 = plot3d_mesh (nc.rows(), els)
plt1
```



Duljine štapova (iz poznatih koordinata čvorova) i sile u štapovima (f = q*l):

```
l = list_of_element_lengths (els, nc)
f = list_of_element_forces (l, qs)
```

```
l
```

```
[0.628138140883318, 0.628138140883314, 0.717831420418702,0.651921444321565,
0.651921444321561, 0.717831420418702,0.859282393771427, 0.731738570212587,
0.671956797913604,0.671956797913600, 0.731738570212586, 0.859282393771427,
1.12548990176379, 0.838216162615248, 0.727607442820430,0.684027538403065,
0.684027538403063, 0.727607442820429,0.838216162615247, itd.]
```

```
f
```

```
[0.628138140883318, 0.628138140883314, 0.717831420418702,0.651921444321565,
0.651921444321561, 0.717831420418702,0.859282393771427, 0.731738570212587,
0.671956797913604,0.671956797913600, 0.731738570212586, 0.859282393771427,
1.12548990176379, 0.838216162615248, 0.727607442820430,0.684027538403065,
0.684027538403063, 0.727607442820429,0.838216162615247, itd.]
```

```
f_bnd = get_values_on_cables (bnd_cbls, tcei, f)
f_bnd
```

```
[14.5444127200577, 14.0435472786861, 13.9171707681621, 14.2083415759035,
14.2083415759035, 13.9171707681621, 14.0435472786861, 14.5444127200577,
14.5444127200577, 14.0435472786861, 13.9171707681621, itd.]
```

```
f_int = get_values_on_cables (int_cbls, tcei, f)
```

```
max (f_int), min (f_int)
```

```
(1.20950803897828, 0.628138140883312)
```

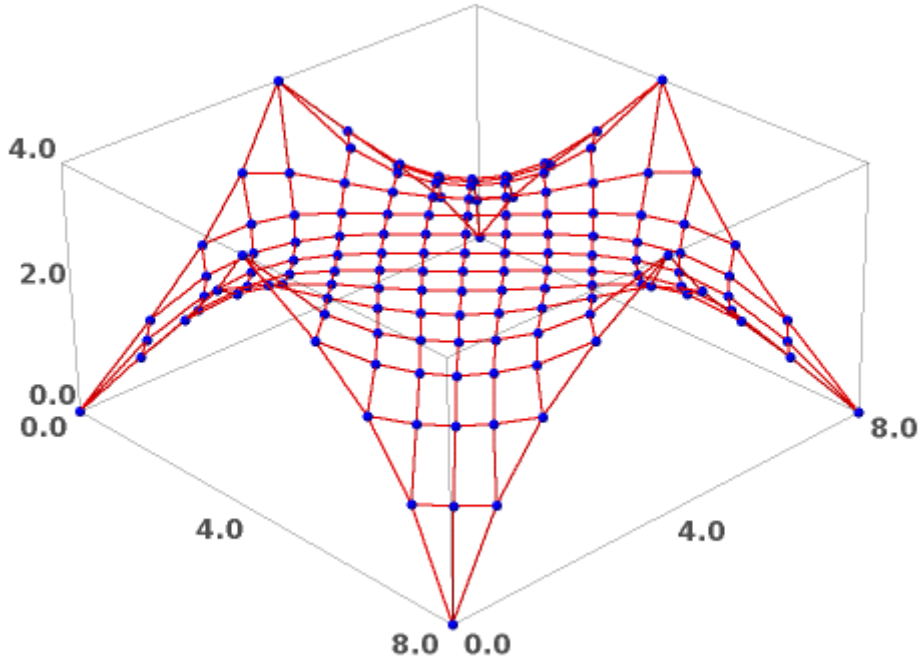
```
max (f_bnd), min (f_bnd)
```

```
(14.5444127200577, 13.9171707681621)
```

```
bels = get_values_on_cables (bnd_cbls, tcei, l)
bels
```

```
[1.45444127200577, 1.40435472786861, 1.39171707681621, 1.42083415759034,
1.42083415759035, 1.39171707681621, 1.40435472786861, itd.]
```

```
nc, f = multistepFDM2 (nds, els, supps, qs, 5)
plot3d_mesh (nc.rows(), els
```




```
l1 = list_of_element_lengths (els, nc.rows())
l1
```

```
[0.561735185231207, 0.561735185231204, 0.677923615357283,0.521237115487068,
0.521237115487067, 0.677923615357285,0.925755845944887, 0.609446384196845,
0.506959325559153,0.506959325559153, 0.609446384196841, 0.925755845944894,
1.46601258881016, 0.790391856348947, 0.587579358820871,0.516457577589277,
0.516457577589279, 0.587579358820869,0.790391856348949, itd.]
```

```
bels = get_values_on_cables (bnd_cbls, tcei, l1)
bels
```

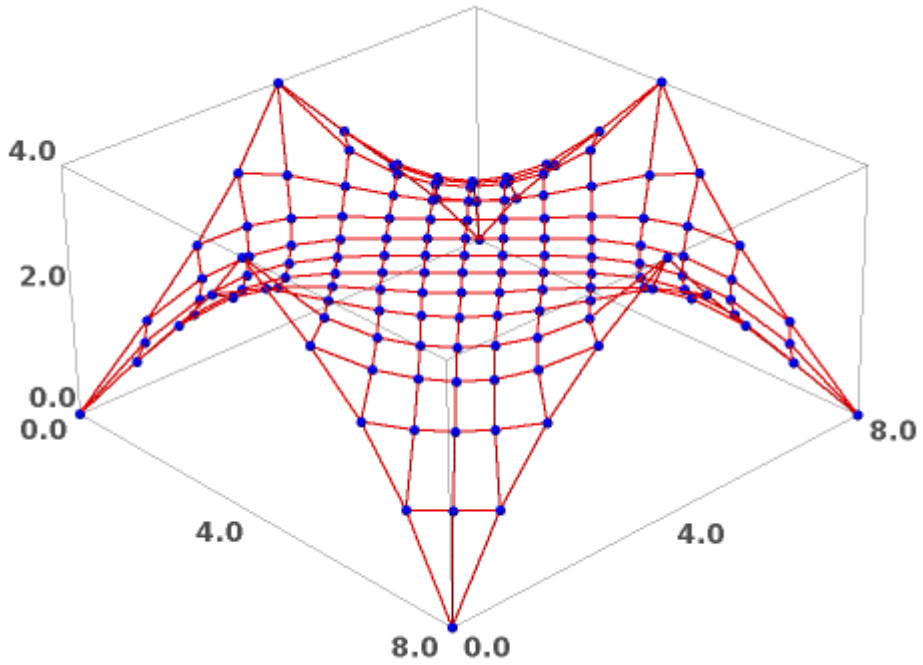
```
[1.68479653084955, 1.32291158500646, 1.23391613680040, 1.49694692274289,
1.49694692274290, 1.23391613680039, 1.32291158500647, itd.]
```

```
beils = zip (beis, bels)
beils
```

```
[(256, 1.68479653084955), (257, 1.32291158500646), (258,1.23391613680040),
(259, 1.49694692274289), (260, 1.49694692274290), itd.]
```

```
ieis = select_elements_on_cables (int_cbls, tcei)
iefs = [1. for i in ieis]
ieifs = zip (ieis, iefs)
```

```
nc, f = multistepFDM (nds, els, supps, qs, ieifs, beils, 10)
plot3d_mesh (nc.rows(), els)
```




```
f
```

```
[0.997518352166359, 0.997518352166362, 1.00364020741154,0.991915979510610,
0.991915979510608, 1.00364020741155,1.00462763662887, 0.999343870376956,
0.989828647070860,0.989828647070865, 0.999343870376954,itd.]
```

```
nc, f = multistepFDM_wtol (nds, els, supps, qs, ieifs, beils,
1.e-4, 1.e-2, 1000)
```

```
steps: 318
maximal force error:  0.0000993544871337182
maximal length error: 6.93945749974390e-7
```

```
nc, f = multistepFDM_wtol (nds, els, supps, qs, ieifs, beils,
1.e-4, 1.e-3, 1000)
```

```
steps: 318
maximal force error:  0.0000993544871337182
maximal length error: 6.93945749974390e-7
```

```
nc, f = multistepFDM_wtol (nds, els, supps, qs, ieifs, beils,
1.e-4, 1.e-4, 2000)
```

```
steps: 318
maximal force error:  0.0000993544871337182
maximal length error: 6.93945749974390e-7
```

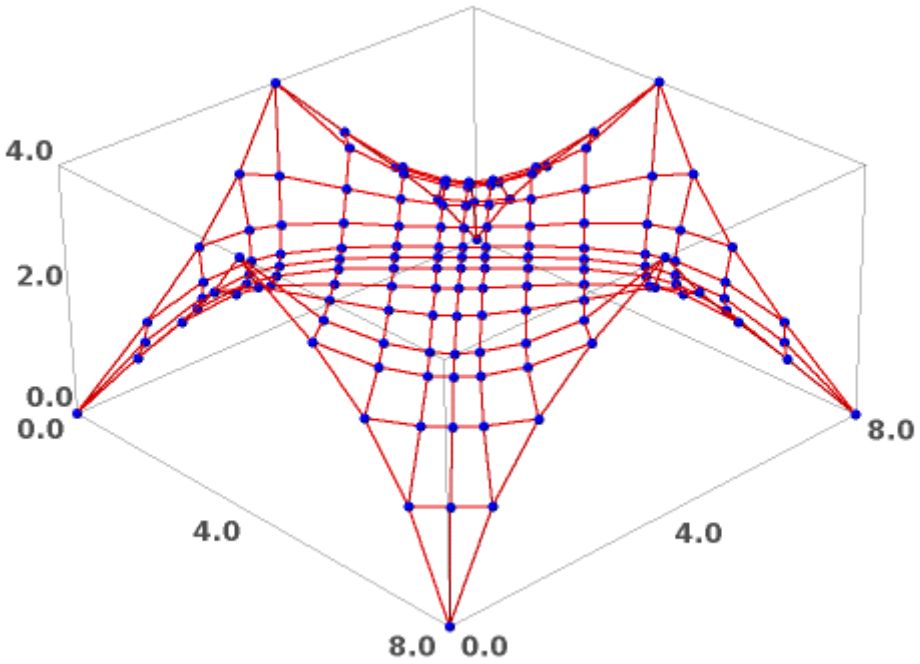
```
f
```

```
[0.999999019333497, 0.999999019333490, 1.00001367217747,0.999974975360994,
0.999974975360994, 1.00001367217746,0.999996675675506, 1.00003080837936,
0.999951653035293,0.999951653035288, 1.00003080837936, itd.]
```

```
f_bnd = get_values_on_cables (bnd_cbls, tcei, f)
max (f_bnd), min (f_bnd)
```

```
(10.6095185132265, 10.0853408121824)
```

```
plot3d_mesh (nc.rows(), els)
```




```
bels
```

```
[1.68479653084955, 1.32291158500646, 1.23391613680040, 1.49694692274289,
1.49694692274290, 1.23391613680039, 1.32291158500647, itd.]
```

```
le1 = mean (bels[0:8])
le2 = mean (bels[8:16])
le3 = mean (bels[16:24])
le4 = mean (bels[24:32])
le1, le2, le3, le4
```

```
(1.43464279384983, 1.43464279384983, 1.43464279384983, 1.43464279384983)
```

```
bels2 = [le1, le1, le1, le1, le2, le2, le2, le2, le3, le3,
le3, le3, le4, le4, le4, le4]


beils2 = zip (beis, bels2)
beils2
```

```
[(256, 1.43464279384983), (257, 1.43464279384983), (258,1.43464279384983),
(259, 1.43464279384983), (260, 1.43464279384983), itd.]
```

```
time nc2, f2 = multistepFDM_wtol (nds, els, supps, qs, ieifs,
beils2, 1.e-4, 1.e-3, 1000)
```

```
steps: 417
maximal force error:  0.0000991170809371278
maximal length error: 1.60476091615180e-6
Time: CPU 241.14 s, Wall: 241.68 s
```

```
f2
```

```
[1.00000034526383, 0.999998297147648, 1.00000882913840, 0.999983110015205,
0.999970881878866, 1.00001025415865, 0.999998076240824, 1.00002083512170,
0.999961332156382, 0.999938647887249, 1.00002312172418, itd.]
```

```
v0 = vector (nc2[128] - nc2[119])
v183 = vector (nc2[111] - nc2[119])
v279 = vector (nc2[136] - nc2[119])
v278 = vector (nc2[102] - nc2[119])
v0, v183, v279, v278
```

```
((0.415104103112747, 0.354752525273011, -0.0302420250739384),
(0.778292153652839, -0.551262731237822, 0.780436274928896),
(-0.223038740720814, 0.985128196552918, -1.04170697423527),
(0.0839796739472156, -1.00526458639603, 0.983908250970747))
```

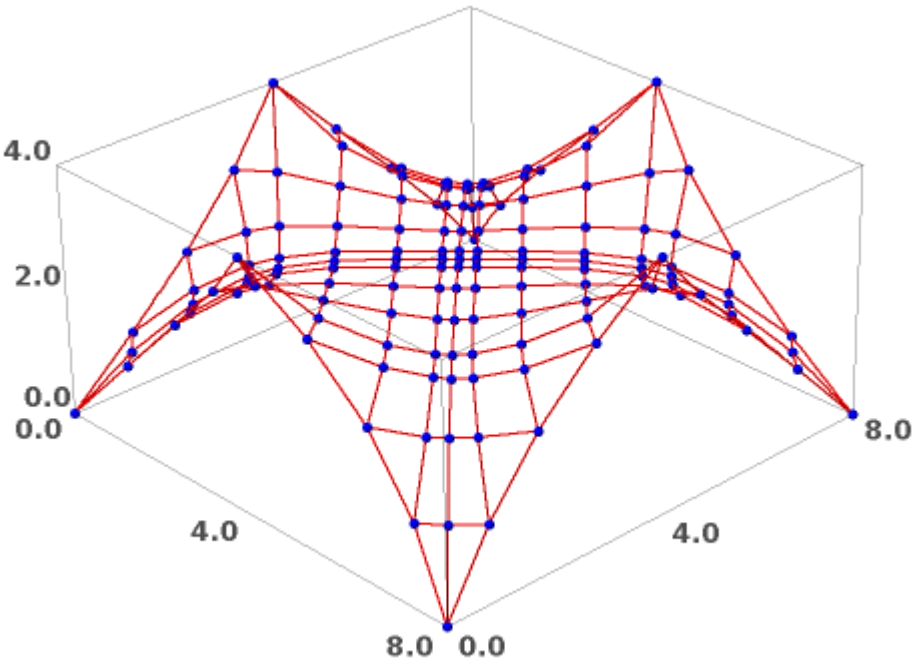
```
f2[0] * v0/norm (v0) + f2[183] * v183/norm (v183) + f2[279] *
v279/norm (v279) + f2[278] * v278/norm (v278)
```

```
(-1.11022302462516e-15, -4.08562073062058e-14, -8.88178419700125e-15)
```

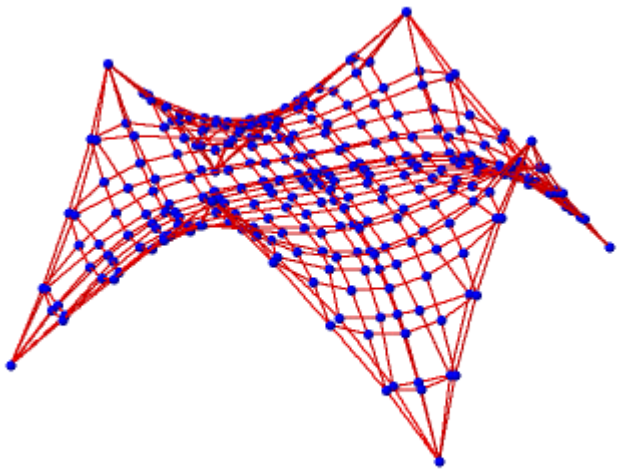
```
f_bnd = get_values_on_cables (bnd_cbls, tcei, f2)
max (f_bnd), min (f_bnd)
```

```
(14.5099182067865, 9.56248161593693)
```

```
plt2 = plot3d_mesh (nc2.rows(), els)
plt2
```



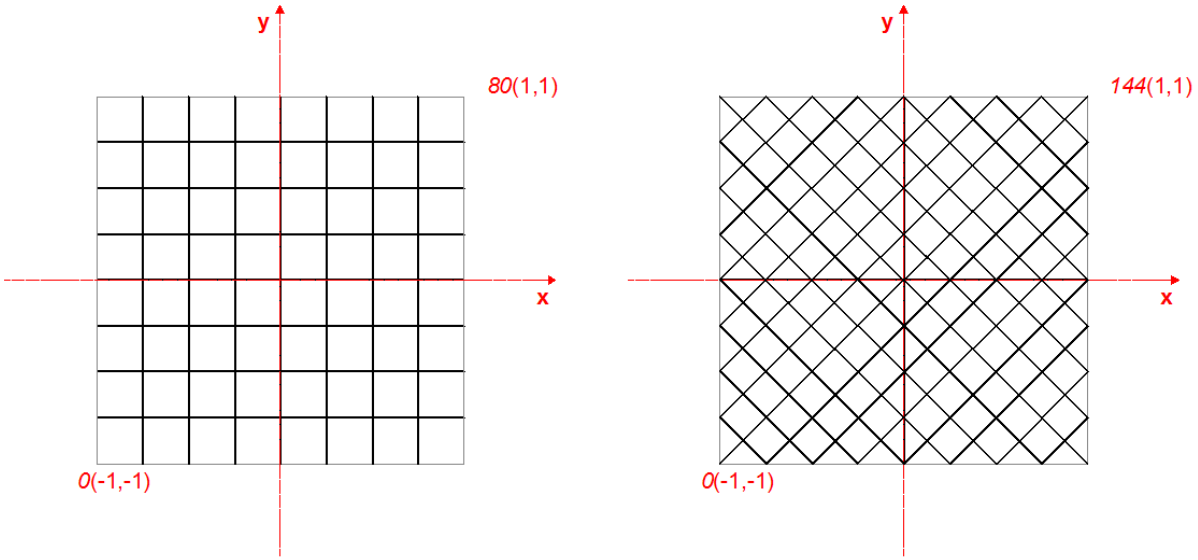

```
(plt1 + plt2).rotateZ (-pi/8).show (frame = False)
```

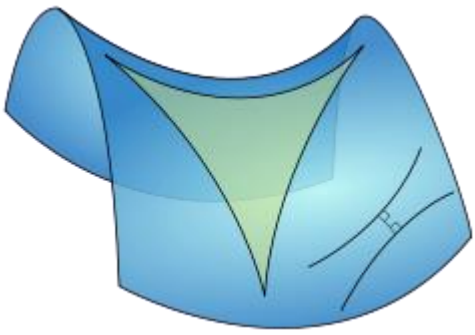


Primjer 4.

U sljedećim primjerima za dva različita oblika mreže i dva različita načina pridržanja tih mreža dobivenih nakon jednog koraka metode gustoća sila uspoređujemo s hiperboličkim paraboloidom koji se zadaje formulom $z=x\cdot y$.

a) b)

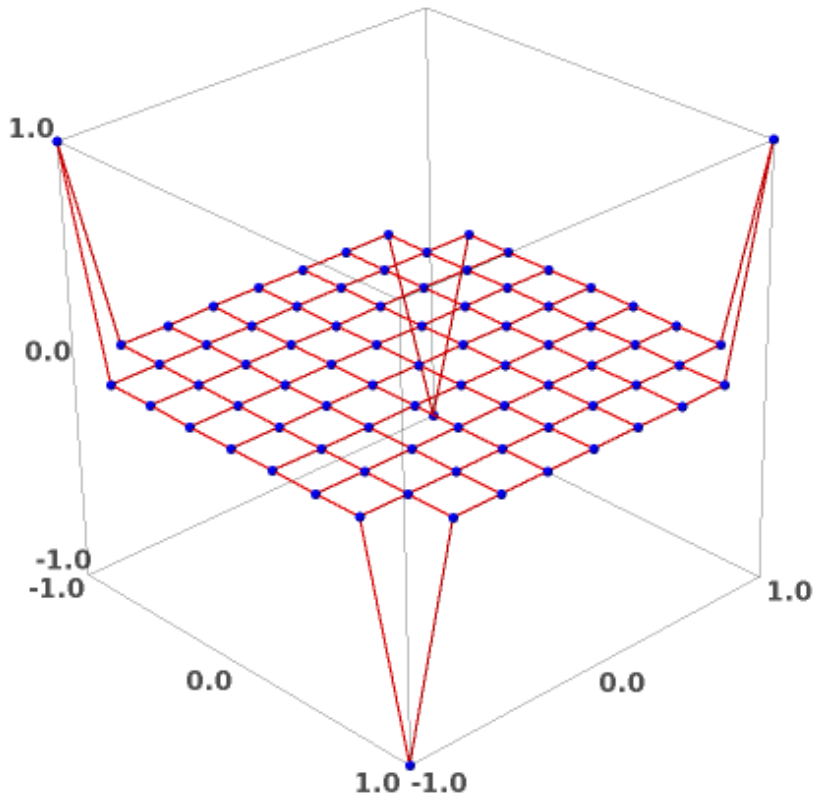




4.1. Mreža a) s rubnim kabelima

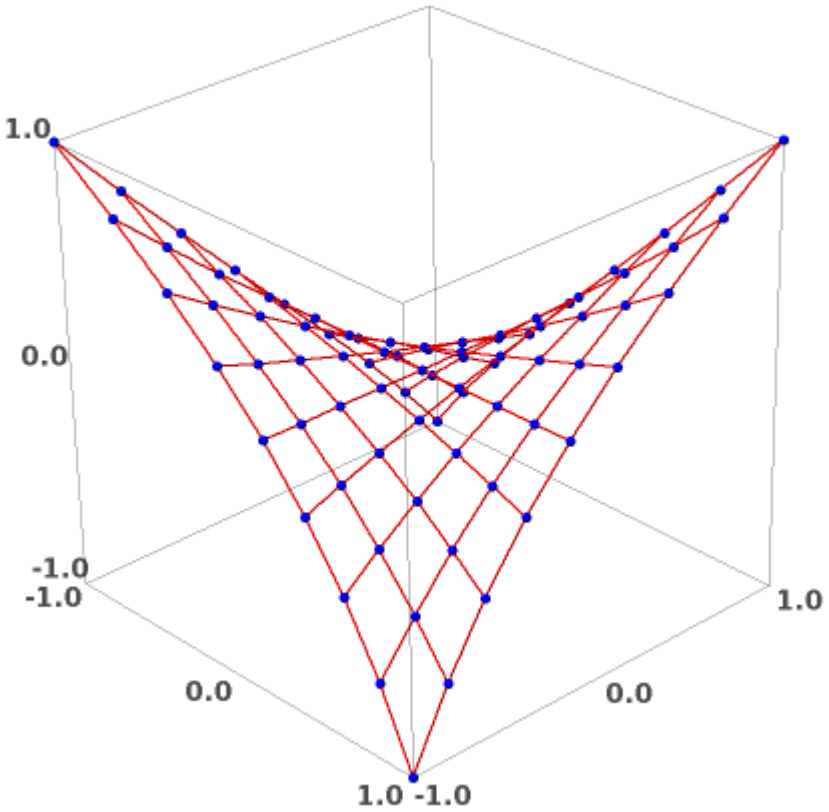
```
nds = make_nodes (9, [-1., -1., 0.], [0.25, 0., 0.])
nds.extend (make_nodes (9, [-1., -0.75, 0.], [0.25, 0., 0.]))
nds.extend (make_nodes (9, [-1., -0.5, 0.], [0.25, 0., 0.]))
nds.extend (make_nodes (9, [-1., -0.25, 0.], [0.25, 0., 0.]))
nds.extend (make_nodes (9, [-1., 0., 0.], [0.25, 0., 0.]))
nds.extend (make_nodes (9, [-1., 0.25, 0.], [0.25, 0., 0.]))
nds.extend (make_nodes (9, [-1., 0.5, 0.], [0.25, 0., 0.]))
nds.extend (make_nodes (9, [-1., 0.75, 0.], [0.25, 0., 0.]))
nds.extend (make_nodes (9, [-1., 1., 0.], [0.25, 0., 0.]))
nds[0] = [-1., -1., 1.]
nds[8] = [1., -1., -1.]
nds[72] = [-1., 1., -1.]
```

```
nds[80] = [1., 1., 1.]

cbls = [cable (63, 9, 1)]
cbls.append (cable (54, 9, 1))
cbls.append (cable (45, 9, 1))
cbls.append (cable (36, 9, 1))
cbls.append (cable (27, 9, 1))
cbls.append (cable (18, 9, 1))
cbls.append (cable (9, 9, 1))
cbls.append (cable (1, 9, 9))
cbls.append (cable (2, 9, 9))
cbls.append (cable (3, 9, 9))
cbls.append (cable (4, 9, 9))
cbls.append (cable (5, 9, 9))
cbls.append (cable (6, 9, 9))
cbls.append (cable (7, 9, 9))
# rubni kabeli:
cbls.append (cable (0, 9, 1))
cbls.append (cable (72, 9, 1))
cbls.append (cable (0, 9, 9))
cbls.append (cable (8, 9, 9))

els = make_elements_on_cables (cbls)
plot3d_mesh(nds,els)
```



```
supps = [0, 8, 72, 80]
tcei = table_of_cable_element_incidences (cbls)
bnd_cbls = [14, 15, 16, 17]
int_cbls = other_cables (bnd_cbls, len (cbls))
beis = select_elements_on_cables (bnd_cbls, tcei)
qs = make_force_densities (len (els))
qs = set_value_on_cables (10., bnd_cbls, tcei, qs)
nc = FDM (nds, els, supps, qs)
plt1 = plot3d_mesh (nc.rows(), els)
```

```
plt1
```



```
print nc.str()
```

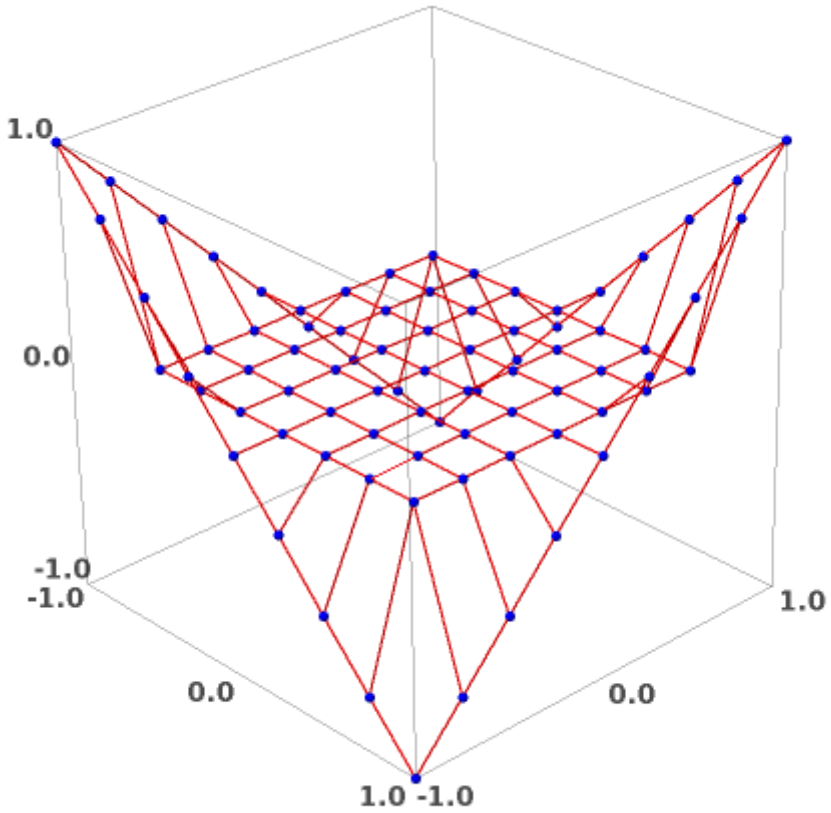
x	y	z	z(r)=x*y	z(r)-z
-1,000	-1,000	1,000	1,000	0,000
-0,746	-0,928	0,729	0,692	-0,037
-0,495	-0,878	0,476	0,435	-0,042
-0,247	-0,848	0,235	0,210	-0,026
0,000	-0,839	0,000	0,000	0,000
0,247	-0,848	-0,235	-0,210	0,026
0,495	-0,878	-0,476	-0,435	0,042
0,746	-0,928	-0,729	-0,692	0,037
1,000	-1,000	-1,000	-1,000	0,000
-0,928	-0,746	0,729	0,692	-0,037
-0,704	-0,704	0,544	0,496	-0,048
-0,472	-0,672	0,360	0,317	-0,042
-0,237	-0,651	0,179	0,154	-0,024
0,000	-0,644	0,000	0,000	0,000
0,237	-0,651	-0,179	-0,154	0,024
0,472	-0,672	-0,360	-0,317	0,042
0,704	-0,704	-0,544	-0,496	0,048
0,928	-0,746	-0,729	-0,692	0,037
-0,878	-0,495	0,476	0,435	-0,042
-0,672	-0,472	0,360	0,317	-0,042
-0,453	-0,453	0,240	0,205	-0,034
-0,228	-0,441	0,119	0,101	-0,019
0,000	-0,437	0,000	0,000	0,000
0,228	-0,441	-0,119	-0,101	0,019
0,453	-0,453	-0,240	-0,205	0,034
0,672	-0,472	-0,360	-0,317	0,042
0,878	-0,495	-0,476	-0,435	0,042
-0,848	-0,247	0,235	0,210	-0,026
-0,651	-0,237	0,179	0,154	-0,024
-0,441	-0,228	0,119	0,101	-0,019
-0,222	-0,222	0,060	0,049	-0,010
0,000	-0,220	0,000	0,000	0,000
0,222	-0,222	-0,060	-0,049	0,010
0,441	-0,228	-0,119	-0,101	0,019
0,651	-0,237	-0,179	-0,154	0,024
0,848	-0,247	-0,235	-0,210	0,026
-0,839	0,000	0,000	0,000	0,000
-0,644	0,000	0,000	0,000	0,000
-0,437	0,000	0,000	0,000	0,000
-0,220	0,000	0,000	0,000	0,000
0,000	0,000	0,000	0,000	0,000
0,220	0,000	0,000	0,000	0,000

0,437	0,000	0,000	0,000	0,000
0,644	0,000	0,000	0,000	0,000
0,839	0,000	0,000	0,000	0,000
-0,848	0,247	-0,235	-0,210	0,026
-0,651	0,237	-0,179	-0,154	0,024
-0,441	0,228	-0,119	-0,101	0,019
-0,222	0,222	-0,060	-0,049	0,010
0,000	0,220	0,000	0,000	0,000
0,222	0,222	0,060	0,049	-0,010
0,441	0,228	0,119	0,101	-0,019
0,651	0,237	0,179	0,154	-0,024
0,848	0,247	0,235	0,210	-0,026
-0,878	0,495	-0,476	-0,435	0,042
-0,672	0,472	-0,360	-0,317	0,042
-0,453	0,453	-0,240	-0,205	0,034
-0,228	0,441	-0,119	-0,101	0,019
0,000	0,437	0,000	0,000	0,000
0,228	0,441	0,119	0,101	-0,019
0,453	0,453	0,240	0,205	-0,034
0,672	0,472	0,360	0,317	-0,042
0,878	0,495	0,476	0,435	-0,042
-0,928	0,746	-0,729	-0,692	0,037
-0,704	0,704	-0,544	-0,496	0,048
-0,472	0,672	-0,360	-0,317	0,042
-0,237	0,651	-0,179	-0,154	0,024
0,000	0,644	0,000	0,000	0,000
0,237	0,651	0,179	0,154	-0,024
0,472	0,672	0,360	0,317	-0,042
0,704	0,704	0,544	0,496	-0,048
0,928	0,746	0,729	0,692	-0,037
-1,000	1,000	-1,000	-1,000	0,000
-0,746	0,928	-0,729	-0,692	0,037
-0,495	0,878	-0,476	-0,435	0,042
-0,247	0,848	-0,235	-0,210	0,026
0,000	0,839	0,000	0,000	0,000
0,247	0,848	0,235	0,210	-0,026
0,495	0,878	0,476	0,435	-0,042
0,746	0,928	0,729	0,692	-0,037
1,000	1,000	1,000	1,000	0,000
maks. odstupanje			+/-	0,048

4.2. Mreža a) s krutim rubovima

```
nds = make_nodes (9, [-1., -1., 0.], [0.25, 0., 0.])
nds.extend (make_nodes (9, [-1., -0.75, 0.], [0.25, 0., 0.]))
nds.extend (make_nodes (9, [-1., -0.5, 0.], [0.25, 0., 0.]))
nds.extend (make_nodes (9, [-1., -0.25, 0.], [0.25, 0., 0.]))
nds.extend (make_nodes (9, [-1., 0., 0.], [0.25, 0., 0.]))
nds.extend (make_nodes (9, [-1., 0.25, 0.], [0.25, 0., 0.]))
nds.extend (make_nodes (9, [-1., 0.5, 0.], [0.25, 0., 0.]))
nds.extend (make_nodes (9, [-1., 0.75, 0.], [0.25, 0., 0.]))
nds.extend (make_nodes (9, [-1., 1., 0.], [0.25, 0., 0.]))

nds[0] = [-1., -1., 1.]
nds[1] = [-0.75, -1., 0.75]
nds[2] = [-0.5, -1., 0.5]
nds[3] = [-0.25, -1., 0.25]
nds[4] = [0., -1., 0.]
nds[5] = [0.25, -1., -0.25]
nds[6] = [0.5, -1., -0.5]
nds[7] = [0.75, -1., -0.75]
nds[8] = [1., -1., -1.]

nds[72] = [-1., 1., -1.]
nds[73] = [-0.75, 1., -0.75]
nds[74] = [-0.5, 1., -0.5]
nds[75] = [-0.25, 1., -0.25]
nds[76] = [0., 1., 0.]
nds[77] = [0.25, 1., 0.25]
nds[78] = [0.5, 1., 0.5]
nds[79] = [0.75, 1., 0.75]
nds[80] = [1., 1., 1.]

nds[9] = [-1., -0.75, 0.75]
nds[18] = [-1., -0.5, 0.5]
nds[27] = [-1., -0.25, 0.25]
nds[36] = [-1., 0., 0.]
nds[45] = [-1., 0.25, -0.25]
nds[54] = [-1., 0.5, -0.5]
nds[63] = [-1., 0.75, -0.75]

nds[17] = [1., -0.75, -0.75]
nds[26] = [1., -0.5, -0.5]
nds[35] = [1., -0.25, -0.25]
nds[44] = [1., 0., 0.]
nds[53] = [1., 0.25, 0.25]
nds[62] = [1., 0.5, 0.5]
nds[71] = [1., 0.75, 0.75]

cbls = [cable (63, 9, 1)]
cbls.append (cable (54, 9, 1))
cbls.append (cable (45, 9, 1))
```

```
cbls.append (cable (36, 9, 1))
cbls.append (cable (27, 9, 1))
cbls.append (cable (18, 9, 1))
cbls.append (cable (9, 9, 1))
cbls.append (cable (1, 9, 9))
cbls.append (cable (2, 9, 9))
cbls.append (cable (3, 9, 9))
cbls.append (cable (4, 9, 9))
cbls.append (cable (5, 9, 9))
cbls.append (cable (6, 9, 9))
cbls.append (cable (7, 9, 9))
# rubni kabeli:
cbls.append (cable (0, 9, 1))
cbls.append (cable (72, 9, 1))
cbls.append (cable (0, 9, 9))
cbls.append (cable (8, 9, 9))

els = make_elements_on_cables (cbls)

plot3d_mesh (nds, els)
```

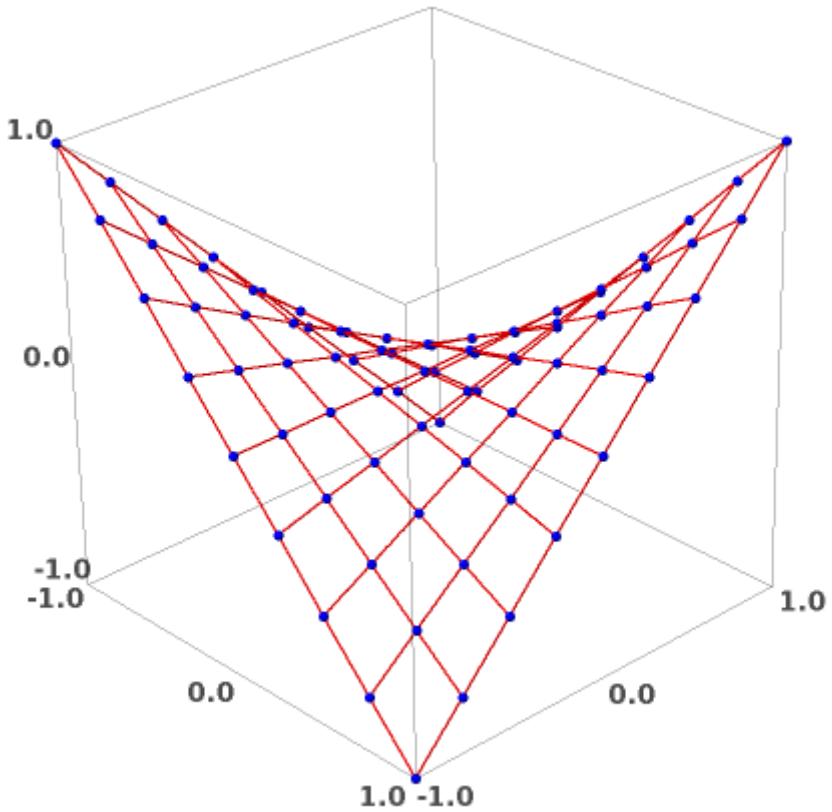



```
supps = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 17, 18, 26, 27, 35, 36,
44, 45, 53, 54, 62, 63, 71, 72, 73, 74, 75, 76, 77, 78, 79,
80]
```

```
tcei = table_of_cable_element_incidences (cbls)

print_indexed (tcei)
```

```
0 :  [0, 1, 2, 3, 4, 5, 6, 7]
1 :  [8, 9, 10, 11, 12, 13, 14, 15]
2 :  [16, 17, 18, 19, 20, 21, 22, 23]
Itd.
```

```
bnd_cbls = [14, 15, 16, 17]

int_cbls = other_cables (bnd_cbls, len (cbls))

beis = select_elements_on_cables (bnd_cbls, tcei)

qs = make_force_densities (len (els))

qs = set_value_on_cables (10., bnd_cbls, tcei, qs)

nc = FDM (nds, els, supps, qs)

plt1 = plot3d_mesh (nc.rows(), els)
plt1
```



```
print nc.str()
```

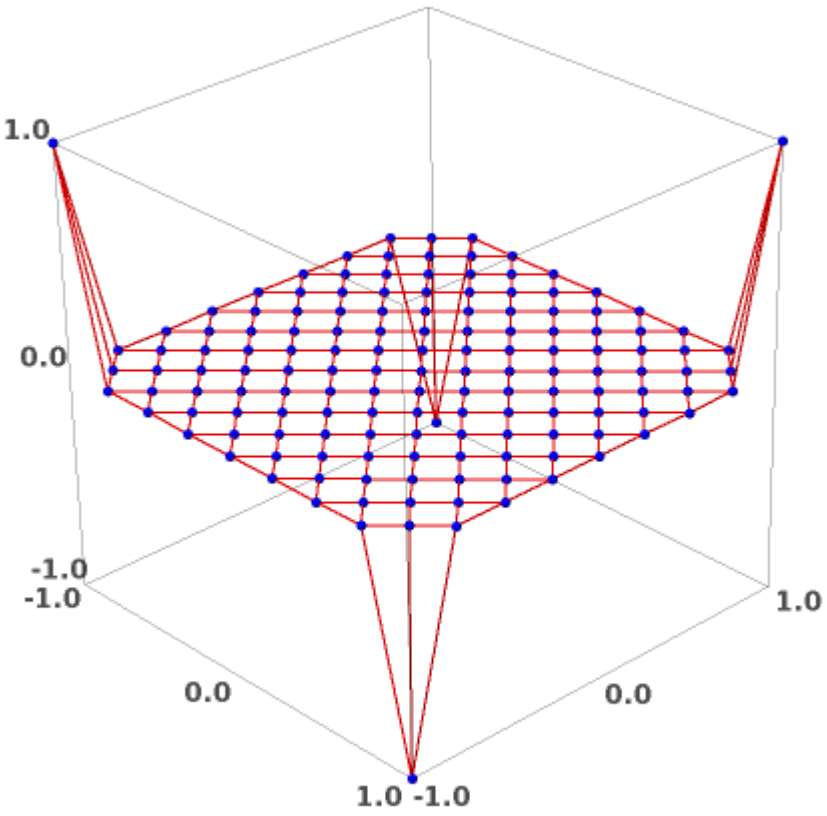
x	y	z	z(r)	z(r)-z
-1,000	-1,000	1,000	1,000	0,000
-0,750	-1,000	0,750	0,750	0,000
-0,500	-1,000	0,500	0,500	0,000
-0,250	-1,000	0,250	0,250	0,000
0,000	-1,000	0,000	0,000	0,000
0,250	-1,000	-0,250	-0,250	0,000
0,500	-1,000	-0,500	-0,500	0,000
0,750	-1,000	-0,750	-0,750	0,000
1,000	-1,000	-1,000	-1,000	0,000
-1,000	-0,750	0,750	0,750	0,000
-0,750	-0,750	0,563	0,563	0,000
-0,500	-0,750	0,375	0,375	0,000
-0,250	-0,750	0,188	0,188	0,000
0,000	-0,750	0,000	0,000	0,000
0,250	-0,750	-0,188	-0,188	0,000
0,500	-0,750	-0,375	-0,375	0,000
0,750	-0,750	-0,563	-0,563	0,000
1,000	-0,750	-0,750	-0,750	0,000
-1,000	-0,500	0,500	0,500	0,000
-0,750	-0,500	0,375	0,375	0,000
-0,500	-0,500	0,250	0,250	0,000
-0,250	-0,500	0,125	0,125	0,000
0,000	-0,500	0,000	0,000	0,000
0,250	-0,500	-0,125	-0,125	0,000
0,500	-0,500	-0,250	-0,250	0,000
0,750	-0,500	-0,375	-0,375	0,000
1,000	-0,500	-0,500	-0,500	0,000
-1,000	-0,250	0,250	0,250	0,000
-0,750	-0,250	0,188	0,188	0,000
-0,500	-0,250	0,125	0,125	0,000
-0,250	-0,250	0,063	0,063	0,000
0,000	-0,250	0,000	0,000	0,000
0,250	-0,250	-0,063	-0,063	0,000
0,500	-0,250	-0,125	-0,125	0,000
0,750	-0,250	-0,188	-0,188	0,000
1,000	-0,250	-0,250	-0,250	0,000
-1,000	0,000	0,000	0,000	0,000
-0,750	0,000	0,000	0,000	0,000
-0,500	0,000	0,000	0,000	0,000
-0,250	0,000	0,000	0,000	0,000
0,000	0,000	0,000	0,000	0,000
0,250	0,000	0,000	0,000	0,000

0,500	0,000	0,000	0,000	0,000
0,750	0,000	0,000	0,000	0,000
1,000	0,000	0,000	0,000	0,000
-1,000	0,250	-0,250	-0,250	0,000
-0,750	0,250	-0,188	-0,188	0,000
-0,500	0,250	-0,125	-0,125	0,000
-0,250	0,250	-0,063	-0,063	0,000
0,000	0,250	0,000	0,000	0,000
0,250	0,250	0,063	0,063	0,000
0,500	0,250	0,125	0,125	0,000
0,750	0,250	0,188	0,188	0,000
1,000	0,250	0,250	0,250	0,000
-1,000	0,500	-0,500	-0,500	0,000
-0,750	0,500	-0,375	-0,375	0,000
-0,500	0,500	-0,250	-0,250	0,000
-0,250	0,500	-0,125	-0,125	0,000
0,000	0,500	0,000	0,000	0,000
0,250	0,500	0,125	0,125	0,000
0,500	0,500	0,250	0,250	0,000
0,750	0,500	0,375	0,375	0,000
1,000	0,500	0,500	0,500	0,000
-1,000	0,750	-0,750	-0,750	0,000
-0,750	0,750	-0,563	-0,563	0,000
-0,500	0,750	-0,375	-0,375	0,000
-0,250	0,750	-0,188	-0,188	0,000
0,000	0,750	0,000	0,000	0,000
0,250	0,750	0,188	0,188	0,000
0,500	0,750	0,375	0,375	0,000
0,750	0,750	0,563	0,563	0,000
1,000	0,750	0,750	0,750	0,000
-1,000	1,000	-1,000	-1,000	0,000
-0,750	1,000	-0,750	-0,750	0,000
-0,500	1,000	-0,500	-0,500	0,000
-0,250	1,000	-0,250	-0,250	0,000
0,000	1,000	0,000	0,000	0,000
0,250	1,000	0,250	0,250	0,000
0,500	1,000	0,500	0,500	0,000
0,750	1,000	0,750	0,750	0,000
1,000	1,000	1,000	1,000	0,000
maks. odstupanje			+/-	0,000

4.3. Mreža b) s rubnim kabelima

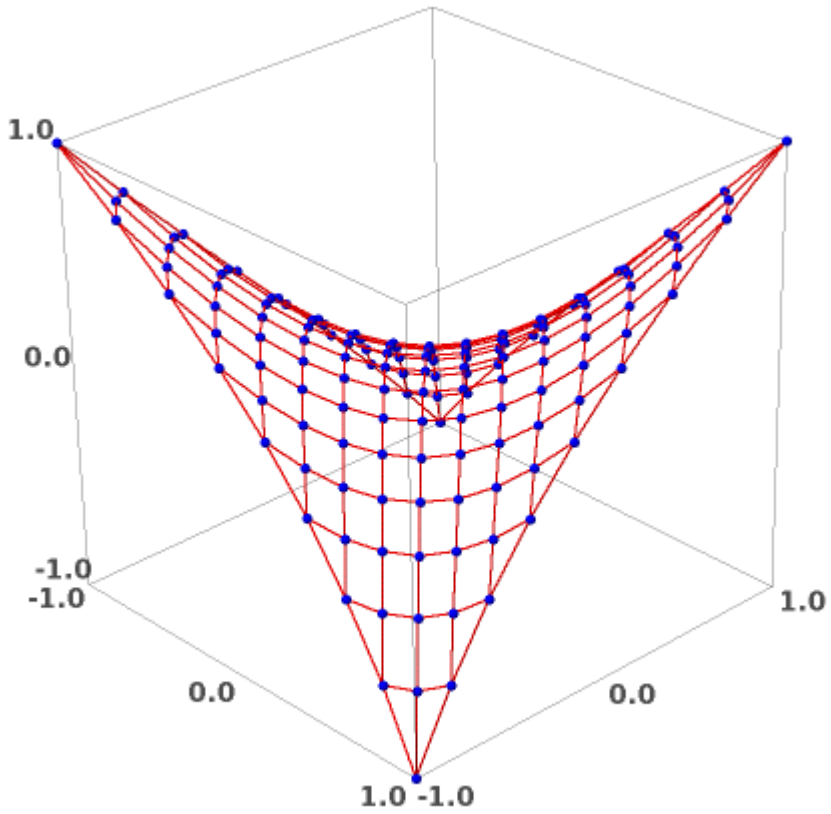
```
nds = make_nodes (9, [-1., -1., 0.], [0.25, 0., 0.])
nds.extend (make_nodes (8, [-0.875, -0.875, 0.], [0.25, 0.,
0.]))
nds.extend (make_nodes (9, [-1., -0.75, 0.], [0.25, 0., 0.]))
nds.extend (make_nodes (8, [-0.875, -0.625, 0.], [0.25, 0.,
0.]))
nds.extend (make_nodes (9, [-1., -0.5, 0.], [0.25, 0., 0.]))
nds.extend (make_nodes (8, [-0.875, -0.375, 0.], [0.25, 0.,
0.]))
nds.extend (make_nodes (9, [-1., -0.25, 0.], [0.25, 0., 0.]))
nds.extend (make_nodes (8, [-0.875,-0.125, 0.], [0.25, 0.,
0.]))
nds.extend (make_nodes (9, [-1., 0., 0.], [0.25, 0., 0.]))
nds.extend (make_nodes (8, [-0.875, 0.125, 0.], [0.25, 0.,
0.]))
nds.extend (make_nodes (9, [-1., 0.25, 0.], [0.25, 0., 0.]))
nds.extend (make_nodes (8, [-0.875, 0.375, 0.], [0.25, 0.,
0.]))
nds.extend (make_nodes (9, [-1., 0.5, 0.], [0.25, 0., 0.]))
nds.extend (make_nodes (8, [-0.875, 0.625, 0.], [0.25, 0.,
0.]))
nds.extend (make_nodes (9, [-1., 0.75, 0.], [0.25, 0., 0.]))
nds.extend (make_nodes (8, [-0.875, 0.875, 0.], [0.25, 0.,
0.]))
nds.extend (make_nodes (9, [-1., 1., 0.], [0.25, 0., 0.]))
nds[0] = [-1., -1., 1.]
nds[8] = [1., -1., -1.]
nds[136] = [-1., 1., -1.]
nds[144] = [1., 1., 1.]

cbls = [cable (119, 3, 9)]
cbls.append (cable (102, 5, 9))
cbls.append (cable (85, 7, 9))
cbls.append (cable (68, 9, 9))
cbls.append (cable (51, 11, 9))
cbls.append (cable (34, 13, 9))
cbls.append (cable (17, 15, 9))
cbls.append (cable (0, 17, 9))
cbls.append (cable (1, 15, 9))
cbls.append (cable (2, 13, 9))
cbls.append (cable (3, 11, 9))
cbls.append (cable (4, 9, 9))
cbls.append (cable (5, 7, 9))
cbls.append (cable (6, 5, 9))
cbls.append (cable (7, 3, 9))
cbls.append (cable (1, 3, 8))
cbls.append (cable (2, 5, 8))
cbls.append (cable (3, 7, 8))
cbls.append (cable (4, 9, 8))
```

```
cbls.append (cable (5, 11, 8))
cbls.append (cable (6, 13, 8))
cbls.append (cable (7, 15, 8))
cbls.append (cable (8, 17, 8))
cbls.append (cable (25, 15, 8))
cbls.append (cable (42, 13, 8))
cbls.append (cable (59, 11, 8))
cbls.append (cable (76, 9, 8))
cbls.append (cable (93, 7, 8))
cbls.append (cable (110, 5, 8))
cbls.append (cable (127, 3, 8))
# rubni kabeli:
cbls.append (cable (0, 9, 1))
cbls.append (cable (136, 9, 1))
cbls.append (cable (0, 9, 17))
cbls.append (cable (8, 9, 17))

els = make_elements_on_cables (cbls)


plot3d_mesh (nds, els)
```



```
supps = [0, 8, 136, 144]

tcei = table_of_cable_element_incidences (cbls)

bnd_cbls = [30, 31, 32, 33]

int_cbls = other_cables (bnd_cbls, len (cbls))

beis = select_elements_on_cables (bnd_cbls, tcei)

qs = make_force_densities (len (els))

qs = set_value_on_cables (10., bnd_cbls, tcei, qs)

nc = FDM (nds, els, supps, qs)

plt1 = plot3d_mesh (nc.rows(), els)
```

```
plt1
```



```
print nc.str()
```

x	y	z	z(r)	z(r)-z
-1,000	-1,000	1,000	1,000	0,000
-0,746	-0,930	0,730	0,694	-0,036
-0,495	-0,882	0,477	0,437	-0,040
-0,247	-0,853	0,236	0,211	-0,025
0,000	-0,844	0,000	0,000	0,000
0,247	-0,853	-0,236	-0,211	0,025
0,495	-0,882	-0,477	-0,437	0,040
0,746	-0,930	-0,730	-0,694	0,036
1,000	-1,000	-1,000	-1,000	0,000
-0,846	-0,846	0,751	0,715	-0,036
-0,605	-0,798	0,529	0,483	-0,045
-0,364	-0,766	0,313	0,279	-0,035
-0,121	-0,750	0,104	0,091	-0,013
0,121	-0,750	-0,104	-0,091	0,013
0,364	-0,766	-0,313	-0,279	0,035
0,605	-0,798	-0,529	-0,483	0,045
0,846	-0,846	-0,751	-0,715	0,036
-0,930	-0,746	0,730	0,694	-0,036
-0,707	-0,707	0,547	0,500	-0,047
-0,474	-0,675	0,361	0,320	-0,041
-0,238	-0,655	0,179	0,156	-0,024
0,000	-0,648	0,000	0,000	0,000
0,238	-0,655	-0,179	-0,156	0,024
0,474	-0,675	-0,361	-0,320	0,041
0,707	-0,707	-0,547	-0,500	0,047
0,930	-0,746	-0,730	-0,694	0,036
-0,798	-0,605	0,529	0,483	-0,045
-0,578	-0,578	0,377	0,334	-0,044
-0,349	-0,557	0,225	0,195	-0,031
-0,117	-0,547	0,075	0,064	-0,011
0,117	-0,547	-0,075	-0,064	0,011
0,349	-0,557	-0,225	-0,195	0,031
0,578	-0,578	-0,377	-0,334	0,044
0,798	-0,605	-0,529	-0,483	0,045
-0,882	-0,495	0,477	0,437	-0,040
-0,675	-0,474	0,361	0,320	-0,041
-0,456	-0,456	0,241	0,208	-0,033
-0,229	-0,443	0,120	0,102	-0,019
0,000	-0,439	0,000	0,000	0,000
0,229	-0,443	-0,120	-0,102	0,019
0,456	-0,456	-0,241	-0,208	0,033

0,675	-0,474	-0,361	-0,320	0,041
0,882	-0,495	-0,477	-0,437	0,040
-0,766	-0,364	0,313	0,279	-0,035
-0,557	-0,349	0,225	0,195	-0,031
-0,338	-0,338	0,135	0,114	-0,021
-0,113	-0,332	0,045	0,038	-0,007
0,113	-0,332	-0,045	-0,038	0,007
0,338	-0,338	-0,135	-0,114	0,021
0,557	-0,349	-0,225	-0,195	0,031
0,766	-0,364	-0,313	-0,279	0,035
-0,853	-0,247	0,236	0,211	-0,025
-0,655	-0,238	0,179	0,156	-0,024
-0,443	-0,229	0,120	0,102	-0,019
-0,224	-0,224	0,060	0,050	-0,010
0,000	-0,222	0,000	0,000	0,000
0,224	-0,224	-0,060	-0,050	0,010
0,443	-0,229	-0,120	-0,102	0,019
0,655	-0,238	-0,179	-0,156	0,024
0,853	-0,247	-0,236	-0,211	0,025
-0,750	-0,121	0,104	0,091	-0,013
-0,547	-0,117	0,075	0,064	-0,011
-0,332	-0,113	0,045	0,038	-0,007
-0,111	-0,111	0,015	0,012	-0,003
0,111	-0,111	-0,015	-0,012	0,003
0,332	-0,113	-0,045	-0,038	0,007
0,547	-0,117	-0,075	-0,064	0,011
0,750	-0,121	-0,104	-0,091	0,013
-0,844	0,000	0,000	0,000	0,000
-0,648	0,000	0,000	0,000	0,000
-0,439	0,000	0,000	0,000	0,000
-0,222	0,000	0,000	0,000	0,000
0,000	0,000	0,000	0,000	0,000
0,222	0,000	0,000	0,000	0,000
0,439	0,000	0,000	0,000	0,000
0,648	0,000	0,000	0,000	0,000
0,844	0,000	0,000	0,000	0,000
-0,750	0,121	-0,104	-0,091	0,013
-0,547	0,117	-0,075	-0,064	0,011
-0,332	0,113	-0,045	-0,038	0,007
-0,111	0,111	-0,015	-0,012	0,003
0,111	0,111	0,015	0,012	-0,003
0,332	0,113	0,045	0,038	-0,007
0,547	0,117	0,075	0,064	-0,011
0,750	0,121	0,104	0,091	-0,013
-0,853	0,247	-0,236	-0,211	0,025

-0,655	0,238	-0,179	-0,156	0,024
-0,443	0,229	-0,120	-0,102	0,019
-0,224	0,224	-0,060	-0,050	0,010
0,000	0,000	0,000	0,000	0,000
0,224	0,224	0,060	0,050	-0,010
0,443	0,229	0,120	0,102	-0,019
0,655	0,238	0,179	0,156	-0,024
0,853	0,247	0,236	0,211	-0,025
-0,766	0,364	-0,313	-0,279	0,035
-0,557	0,349	-0,225	-0,195	0,031
-0,338	0,338	-0,135	-0,114	0,021
-0,113	0,332	-0,045	-0,038	0,007
0,113	0,332	0,045	0,038	-0,007
0,338	0,338	0,135	0,114	-0,021
0,557	0,349	0,225	0,195	-0,031
0,766	0,364	0,313	0,279	-0,035
-0,882	0,495	-0,477	-0,437	0,040
-0,675	0,474	-0,361	-0,320	0,041
-0,456	0,456	-0,241	-0,208	0,033
-0,229	0,443	-0,120	-0,102	0,019
0,000	0,439	0,000	0,000	0,000
0,229	0,443	0,120	0,102	-0,019
0,456	0,456	0,241	0,208	-0,033
0,675	0,474	0,361	0,320	-0,041
0,882	0,495	0,477	0,437	-0,040
-0,798	0,605	-0,529	-0,483	0,045
-0,578	0,578	-0,377	-0,334	0,044
-0,349	0,557	-0,225	-0,195	0,031
-0,117	0,547	-0,075	-0,064	0,011
0,117	0,547	0,075	0,064	-0,011
0,349	0,557	0,225	0,195	-0,031
0,578	0,578	0,377	0,334	-0,044
0,798	0,605	0,529	0,483	-0,045
-0,930	0,746	-0,730	-0,694	0,036
-0,707	0,707	-0,547	-0,500	0,047
-0,474	0,675	-0,361	-0,320	0,041
-0,238	0,655	-0,179	-0,156	0,024
0,000	0,648	0,000	0,000	0,000
0,238	0,655	0,179	0,156	-0,024
0,474	0,675	0,361	0,320	-0,041
0,707	0,707	0,547	0,500	-0,047
0,930	0,746	0,730	0,694	-0,036
-0,846	0,846	-0,751	-0,715	0,036
-0,605	0,798	-0,529	-0,483	0,045
-0,364	0,766	-0,313	-0,279	0,035

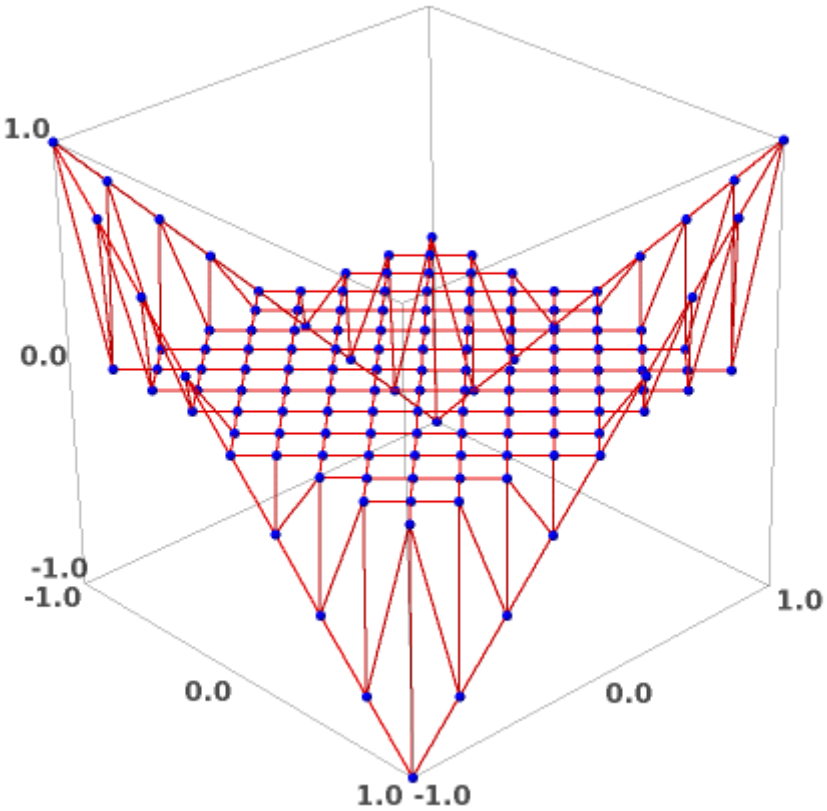
-0,121	0,750	-0,104	-0,091	0,013
0,121	0,750	0,104	0,091	-0,013
0,364	0,766	0,313	0,279	-0,035
0,605	0,798	0,529	0,483	-0,045
0,846	0,846	0,751	0,715	-0,036
-1,000	1,000	-1,000	-1,000	0,000
-0,746	0,930	-0,730	-0,694	0,036
-0,495	0,882	-0,477	-0,437	0,040
-0,247	0,853	-0,236	-0,211	0,025
0,000	0,844	0,000	0,000	0,000
0,247	0,853	0,236	0,211	-0,025
0,495	0,882	0,477	0,437	-0,040
0,746	0,930	0,730	0,694	-0,036
1,000	1,000	1,000	1,000	0,000
maks. odstupanje			+/-	0,047

4.4. Mreža b) s krutim kabelima

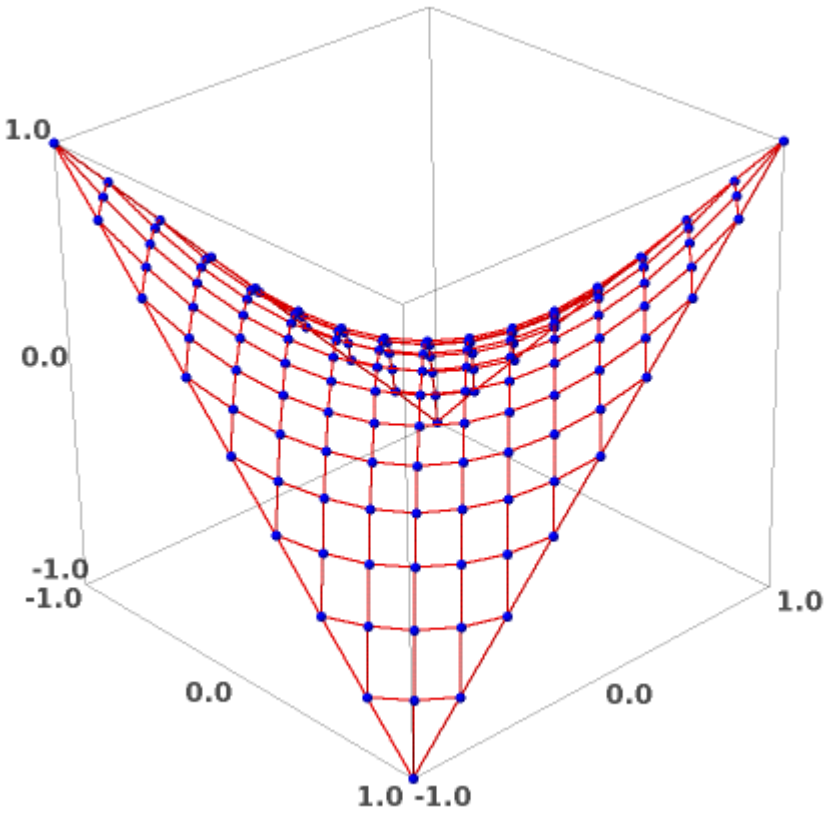
```
nds = make_nodes (9, [-1., -1., 0.], [0.25, 0., 0.])
nds.extend (make_nodes (8, [-0.875, -0.875, 0.], [0.25, 0.,
0.]))
nds.extend (make_nodes (9, [-1., -0.75, 0.], [0.25, 0., 0.]))
nds.extend (make_nodes (8, [-0.875, -0.625, 0.], [0.25, 0.,
0.]))
nds.extend (make_nodes (9, [-1., -0.5, 0.], [0.25, 0., 0.]))
nds.extend (make_nodes (8, [-0.875, -0.375, 0.], [0.25, 0.,
0.]))
nds.extend (make_nodes (9, [-1., -0.25, 0.], [0.25, 0., 0.]))
nds.extend (make_nodes (8, [-0.875,-0.125, 0.], [0.25, 0.,
0.]))
nds.extend (make_nodes (9, [-1., 0., 0.], [0.25, 0., 0.]))
nds.extend (make_nodes (8, [-0.875, 0.125, 0.], [0.25, 0.,
0.]))
nds.extend (make_nodes (9, [-1., 0.25, 0.], [0.25, 0., 0.]))
nds.extend (make_nodes (8, [-0.875, 0.375, 0.], [0.25, 0.,
0.]))
nds.extend (make_nodes (9, [-1., 0.5, 0.], [0.25, 0., 0.]))
nds.extend (make_nodes (8, [-0.875, 0.625, 0.], [0.25, 0.,
0.]))
nds.extend (make_nodes (9, [-1., 0.75, 0.], [0.25, 0., 0.]))
nds.extend (make_nodes (8, [-0.875, 0.875, 0.], [0.25, 0.,
0.]))
nds.extend (make_nodes (9, [-1., 1., 0.], [0.25, 0., 0.]))

nds[0] = [-1., -1., 1.]
nds[1] = [-0.75, -1., 0.75]
nds[2] = [-0.5, -1., 0.5]
```

```
nds[3] = [-0.25, -1., 0.25]
nds[4] = [0., -1., 0.]
nds[5] = [0.25, -1., -0.25]
nds[6] = [0.5, -1., -0.5]
nds[7] = [0.75, -1., -0.75]
nds[8] = [1., -1., -1.]

nds[136] = [-1., 1., -1.]
nds[137] = [-0.75, 1., -0.75]
nds[138] = [-0.5, 1., -0.5]
nds[139] = [-0.25, 1., -0.25]
nds[140] = [0., 1., 0.]
nds[141] = [0.25, 1., 0.25]
nds[142] = [0.5, 1., 0.5]
nds[143] = [0.75, 1., 0.75]
nds[144] = [1., 1., 1.]

nds[17] = [-1., -0.75, 0.75]
nds[34] = [-1., -0.5, 0.5]
nds[51] = [-1., -0.25, 0.25]
nds[68] = [-1., 0., 0.]
nds[85] = [-1., 0.25, -0.25]
nds[102] = [-1., 0.5, -0.5]
nds[119] = [-1., 0.75, -0.75]

nds[25] = [1., -0.75, -0.75]
nds[42] = [1., -0.5, -0.5]
nds[59] = [1., -0.25, -0.25]
nds[76] = [1., 0., 0.]
nds[93] = [1., 0.25, 0.25]
nds[110] = [1., 0.5, 0.5]
nds[127] = [1., 0.75, 0.75]

cbls = [cable (119, 3, 9)]
cbls.append (cable (102, 5, 9))
cbls.append (cable (85, 7, 9))
cbls.append (cable (68, 9, 9))
cbls.append (cable (51, 11, 9))
cbls.append (cable (34, 13, 9))
cbls.append (cable (17, 15, 9))
cbls.append (cable (0, 17, 9))
cbls.append (cable (1, 15, 9))
cbls.append (cable (2, 13, 9))
cbls.append (cable (3, 11, 9))
cbls.append (cable (4, 9, 9))
cbls.append (cable (5, 7, 9))
cbls.append (cable (6, 5, 9))
cbls.append (cable (7, 3, 9))
cbls.append (cable (1, 3, 8))
```

```
cbls.append (cable (2, 5, 8))
cbls.append (cable (3, 7, 8))
cbls.append (cable (4, 9, 8))
cbls.append (cable (5, 11, 8))
cbls.append (cable (6, 13, 8))
cbls.append (cable (7, 15, 8))
cbls.append (cable (8, 17, 8))
cbls.append (cable (25, 15, 8))
cbls.append (cable (42, 13, 8))
cbls.append (cable (59, 11, 8))
cbls.append (cable (76, 9, 8))
cbls.append (cable (93, 7, 8))
cbls.append (cable (110, 5, 8))
cbls.append (cable (127, 3, 8))
# rubni kabeli:
cbls.append (cable (0, 9, 1))
cbls.append (cable (136, 9, 1))
cbls.append (cable (0, 9, 17))
cbls.append (cable (8, 9, 17))
els = make_elements_on_cables (cbls)

plot3d_mesh (nds, els)
```



```
supps = [0, 1, 2, 3, 4, 5, 6, 7, 8, 17, 25, 34, 42, 51, 59,
68, 76, 85, 93, 102, 110, 119, 127, 136, 137, 138, 139, 140,
141, 142, 143, 144]

tcei = table_of_cable_element_incidences (cbls)

bnd_cbls = [30, 31, 32, 33]

int_cbls = other_cables (bnd_cbls, len (cbls))

beis = select_elements_on_cables (bnd_cbls, tcei)

qs = make_force_densities (len (els))

qs = set_value_on_cables (10., bnd_cbls, tcei, qs)

nc = FDM (nds, els, supps, qs)

plt1 = plot3d_mesh (nc.rows(), els)


plt1
```



```
print nc.str()
```

x	y	z	z(r)	z(r)-z
-1,000	-1,000	1,000	1,000	0,000
-0,750	-1,000	0,750	0,750	0,000
-0,500	-1,000	0,500	0,500	0,000
-0,250	-1,000	0,250	0,250	0,000
0,000	-1,000	0,000	0,000	0,000
0,250	-1,000	-0,250	-0,250	0,000
0,500	-1,000	-0,500	-0,500	0,000
0,750	-1,000	-0,750	-0,750	0,000
1,000	-1,000	-1,000	-1,000	0,000
-0,875	-0,875	0,766	0,766	0,000
-0,625	-0,875	0,547	0,547	0,000
-0,375	-0,875	0,328	0,328	0,000
-0,125	-0,875	0,109	0,109	0,000
0,125	-0,875	-0,109	-0,109	0,000
0,375	-0,875	-0,328	-0,328	0,000
0,625	-0,875	-0,547	-0,547	0,000
0,875	-0,875	-0,766	-0,766	0,000
-1,000	-0,750	0,750	0,750	0,000
-0,750	-0,750	0,562	0,562	0,000
-0,500	-0,750	0,375	0,375	0,000
-0,250	-0,750	0,188	0,188	0,000
0,000	-0,750	0,000	0,000	0,000
0,250	-0,750	-0,188	-0,188	0,000
0,500	-0,750	-0,375	-0,375	0,000
0,750	-0,750	-0,563	-0,563	0,000
1,000	-0,750	-0,750	-0,750	0,000
-0,875	-0,625	0,547	0,547	0,000
-0,625	-0,625	0,391	0,391	0,000
-0,375	-0,625	0,234	0,234	0,000
-0,125	-0,625	0,078	0,078	0,000
0,125	-0,625	-0,078	-0,078	0,000
0,375	-0,625	-0,234	-0,234	0,000
0,625	-0,625	-0,391	-0,391	0,000
0,875	-0,625	-0,547	-0,547	0,000
-1,000	-0,500	0,500	0,500	0,000
-0,750	-0,500	0,375	0,375	0,000
-0,500	-0,500	0,250	0,250	0,000
-0,250	-0,500	0,125	0,125	0,000
0,000	-0,500	0,000	0,000	0,000
0,250	-0,500	-0,125	-0,125	0,000
0,500	-0,500	-0,250	-0,250	0,000
0,750	-0,500	-0,375	-0,375	0,000

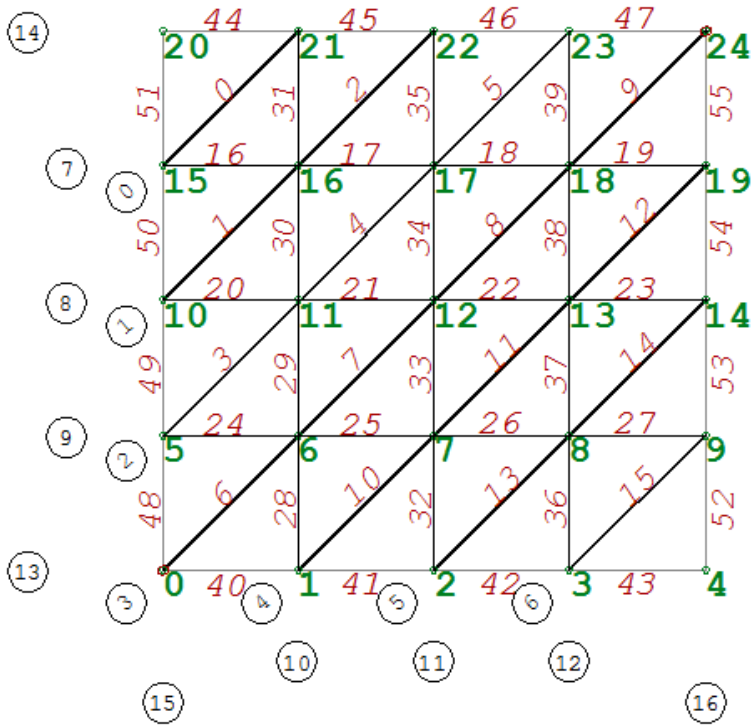
1,000	-0,500	-0,500	-0,500	0,000
-0,875	-0,375	0,328	0,328	0,000
-0,625	-0,375	0,234	0,234	0,000
-0,375	-0,375	0,141	0,141	0,000
-0,125	-0,375	0,047	0,047	0,000
0,125	-0,375	-0,047	-0,047	0,000
0,375	-0,375	-0,141	-0,141	0,000
0,625	-0,375	-0,234	-0,234	0,000
0,875	-0,375	-0,328	-0,328	0,000
-1,000	-0,250	0,250	0,250	0,000
-0,750	-0,250	0,188	0,188	0,000
-0,500	-0,250	0,125	0,125	0,000
-0,250	-0,250	0,062	0,063	0,000
0,000	-0,250	0,000	0,000	0,000
0,250	-0,250	-0,063	-0,063	0,000
0,500	-0,250	-0,125	-0,125	0,000
0,750	-0,250	-0,188	-0,188	0,000
1,000	-0,250	-0,250	-0,250	0,000
-0,875	-0,125	0,109	0,109	0,000
-0,625	-0,125	0,078	0,078	0,000
-0,375	-0,125	0,047	0,047	0,000
-0,125	-0,125	0,016	0,016	0,000
0,125	-0,125	-0,016	-0,016	0,000
0,375	-0,125	-0,047	-0,047	0,000
0,625	-0,125	-0,078	-0,078	0,000
0,875	-0,125	-0,109	-0,109	0,000
-1,000	0,000	0,000	0,000	0,000
-0,750	0,000	0,000	0,000	0,000
-0,500	0,000	0,000	0,000	0,000
-0,250	0,000	0,000	0,000	0,000
0,000	0,000	0,000	0,000	0,000
0,250	0,000	0,000	0,000	0,000
0,500	0,000	0,000	0,000	0,000
0,750	0,000	0,000	0,000	0,000
1,000	0,000	0,000	0,000	0,000
-0,875	0,125	-0,109	-0,109	0,000
-0,625	0,125	-0,078	-0,078	0,000
-0,375	0,125	-0,047	-0,047	0,000
-0,125	0,125	-0,016	-0,016	0,000
0,125	0,125	0,016	0,016	0,000
0,375	0,125	0,047	0,047	0,000
0,625	0,125	0,078	0,078	0,000
0,875	0,125	0,109	0,109	0,000
-1,000	0,250	-0,250	-0,250	0,000
-0,750	0,250	-0,188	-0,188	0,000

-0,500	0,250	-0,125	-0,125	0,000
-0,250	0,250	-0,063	-0,063	0,000
0,000	0,250	0,000	0,000	0,000
0,250	0,250	0,063	0,063	0,000
0,500	0,250	0,125	0,125	0,000
0,750	0,250	0,188	0,188	0,000
1,000	0,250	0,250	0,250	0,000
-0,875	0,375	-0,328	-0,328	0,000
-0,625	0,375	-0,234	-0,234	0,000
-0,375	0,375	-0,141	-0,141	0,000
-0,125	0,375	-0,047	-0,047	0,000
0,125	0,375	0,047	0,047	0,000
0,375	0,375	0,141	0,141	0,000
0,625	0,375	0,234	0,234	0,000
0,875	0,375	0,328	0,328	0,000
-1,000	0,500	-0,500	-0,500	0,000
-0,750	0,500	-0,375	-0,375	0,000
-0,500	0,500	-0,250	-0,250	0,000
-0,250	0,500	-0,125	-0,125	0,000
0,000	0,500	0,000	0,000	0,000
0,250	0,500	0,125	0,125	0,000
0,500	0,500	0,250	0,250	0,000
0,750	0,500	0,375	0,375	0,000
1,000	0,500	0,500	0,500	0,000
-0,875	0,625	-0,547	-0,547	0,000
-0,625	0,625	-0,391	-0,391	0,000
-0,375	0,625	-0,234	-0,234	0,000
-0,125	0,625	-0,078	-0,078	0,000
0,125	0,625	0,078	0,078	0,000
0,375	0,625	0,234	0,234	0,000
0,625	0,625	0,391	0,391	0,000
0,875	0,625	0,547	0,547	0,000
-1,000	0,750	-0,750	-0,750	0,000
-0,750	0,750	-0,563	-0,563	0,000
-0,500	0,750	-0,375	-0,375	0,000
-0,250	0,750	-0,188	-0,188	0,000
0,000	0,750	0,000	0,000	0,000
0,250	0,750	0,188	0,188	0,000
0,500	0,750	0,375	0,375	0,000
0,750	0,750	0,563	0,563	0,000
1,000	0,750	0,750	0,750	0,000
-0,875	0,875	-0,766	-0,766	0,000
-0,625	0,875	-0,547	-0,547	0,000
-0,375	0,875	-0,328	-0,328	0,000
-0,125	0,875	-0,109	-0,109	0,000

0,125	0,875	0,109	0,109	0,000
0,375	0,875	0,328	0,328	0,000
0,625	0,875	0,547	0,547	0,000
0,875	0,875	0,766	0,766	0,000
-1,000	1,000	-1,000	-1,000	0,000
-0,750	1,000	-0,750	-0,750	0,000
-0,500	1,000	-0,500	-0,500	0,000
-0,250	1,000	-0,250	-0,250	0,000
0,000	1,000	0,000	0,000	0,000
0,250	1,000	0,250	0,250	0,000
0,500	1,000	0,500	0,500	0,000
0,750	1,000	0,750	0,750	0,000
1,000	1,000	1,000	1,000	0,000
maks. odstupanje			+/-	0,000

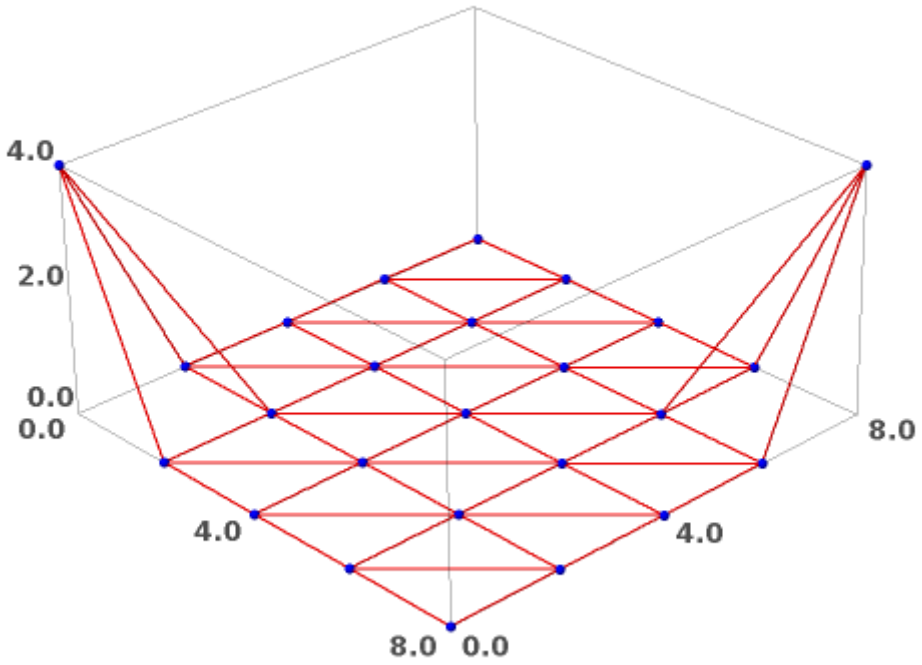
Primjer 5.

U ovom primjeru analizirat ćemo mrežu s tri familije užadi.




```
nds = make_nodes (5, [0., 0., 0.], [2., 0., 0.])
nds.extend (make_nodes (5, [0., 2., 0.], [2., 0., 0.]))
nds.extend (make_nodes (5, [0., 4., 0.], [2., 0., 0.]))
nds.extend (make_nodes (5, [0., 6., 0.], [2., 0., 0.]))
nds.extend (make_nodes (5, [0., 8., 0.], [2., 0., 0.]))
nds[0] = [0., 0., 4.]
nds[24] = [8., 8., 4.]

cbls = [cable (15, 2, 6)]
cbls.append (cable (10, 3, 6))
cbls.append (cable (5, 4, 6))
cbls.append (cable (0, 5, 6))
cbls.append (cable (1, 4, 6))
cbls.append (cable (2, 3, 6))
cbls.append (cable (3, 2, 6))

cbls.append (cable (15, 5, 1))
cbls.append (cable (10, 5, 1))
cbls.append (cable (5, 5, 1))
cbls.append (cable (1, 5, 5))
cbls.append (cable (2, 5, 5))
cbls.append (cable (3, 5, 5))
# rubni kabeli:
cbls.append (cable (0, 5, 1))
cbls.append (cable (20, 5, 1))
```

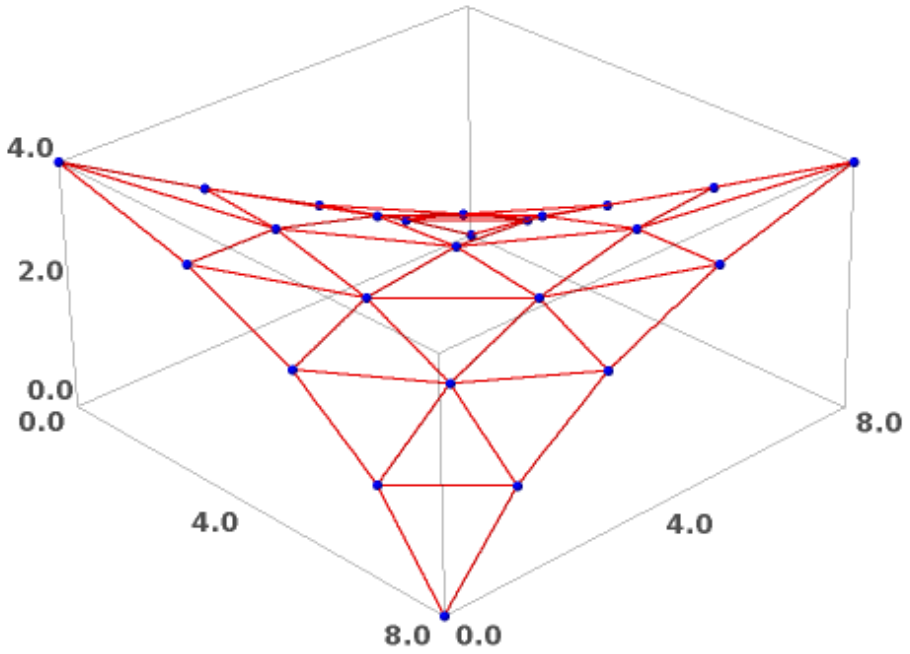
```
cbls.append (cable (0, 5, 5))
cbls.append (cable (4, 5, 5))
els = make_elements_on_cables (cbls)

plot3d_mesh (nds, els)
```



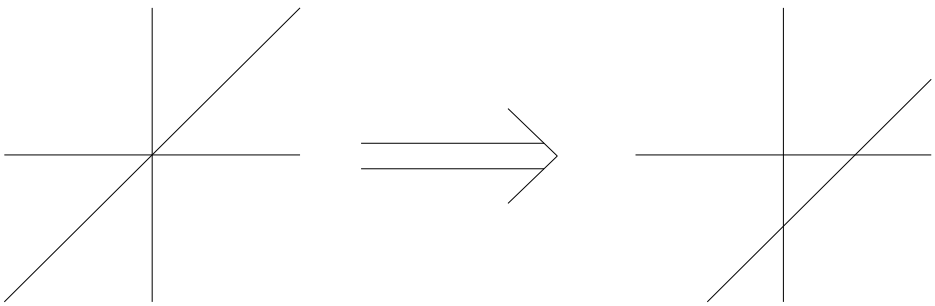

```
supps = [0, 4, 20, 24]


tcei = table_of_cable_element_incidences (cbls)
print_indexed (tcei)
```

```
0 :  [0]
1 :  [1, 2]
2 :  [3, 4, 5]
3 :  [6, 7, 8, 9]
4 :  [10, 11, 12]
5 :  [13, 14]
6 :  [15]
7 :  [16, 17, 18, 19]
8 :  [20, 21, 22, 23]
9 :  [24, 25, 26, 27]
10 :  [28, 29, 30, 31]
11 :  [32, 33, 34, 35]
12 :  [36, 37, 38, 39]
13 :  [40, 41, 42, 43]
14 :  [44, 45, 46, 47]
15 :  [48, 49, 50, 51]
16 :  [52, 53, 54, 55]
```

```
bnd_cbls = [13, 14, 15, 16]
int_cbls = other_cables (bnd_cbls, len (cbls))
beis = select_elements_on_cables (bnd_cbls, tcei)

qs = make_force_densities (len (els))

qs = set_value_on_cables (10., bnd_cbls, tcei, qs)

nc = FDM (nds, els, supps, qs)

plt1 = plot3d_mesh (nc.rows(), els)
plt1
```



Ako se uzme da se u čvoru sijeku tri kabela, ne može se dobiti minimalna mreža, jer dopusti li se klizanje kabela, treći kabel neće proći kroz sjecište ostala dva.



5. Zaključak

Budući da se konstrukcije od užadi bitno razlikuju od tradicionalnih konstrukcija, trebale su se pronaći i nove metode kojima bi se omogućilo njihovo projektiranje, a samim time funkcionalsnost i stabilnost. U početku su se jednostavniji oblici konstrukcija od užadi nalazili u prirodi, međutim nailaskom na složenije i zahtjevnije probleme trebala se minucioznije istražiti sama bit te ponašanja takvih konstrukcija. Glavna prednost užeta je prenošenjaje opterećenja vlačnom silom čime se ostvaruje bolja iskorištenost poprečnih presjeka i samim time prekoračivanje većih udaljenosti bez dodatnih oslonaca. Atraktivnost različitih oblika, koji ujedno imaju i funkcionalne uluge, učinila je takve konstrukcije pravim umjetničkim djelima i povećala njihovu popularnost i primjenu, budući da takve konstrukcije ostavljaju upečatljiv dojam na gledatelja. Specifičnost, koja je proučena u ovom radu, leži u činjenici da se oblik ne može nametnuti, već se mora pronaći kao međuzavisnost raznih paramatara. Proces *form finding*-a, te statičke i dinamičke analize danas je gotovo nezamisliv bez primjene računala, koje omogućuje veće brzine i bolju preciznost, uz uvažavanje svih parametara, vodeći ka optimalnom obliku.

U primjerima je prikazan slijed postupaka traženja oblika kad na konstrukciju ne djeluje optrećenje. Mijenjajući geometriju mreže, pridržanja i vrijednosti sila u pojedinim kabelima dobili smo različite oblike ravnotežnog stanja mreže. Dobivene mreže su uspoređivane s minimalnima, da bi se vidjela odstupanja, budući da minimalna mreža predstavlja energijski optimum. Kasnije je potrebno ispitati zadovoljava li oblik dobiven jednim od tih ograničenja zahtjeve arhitekta i konstruktora, te, ako je potrebno, postupak ponoviti, što zajedno čini proces optimizacije konstrukcije.

6. Literatura

1) Santoso, Katherina.: Wide-Span Cable Structures, University of California, Barkley, 2003.

2) Dvornik, J.; Lazarević,D.: Prednapete gipke konstrukcije od užadi i tkanine, Građevinar, 47(1995)4, str. 185-199.

3) Gidak, P.: Primjena metode gustoće sila na oblikovanje prednapetih mreža, magistarski rad

4) Fresl, K.; Gidak, P.: Prednapete gipke konstrukcije od užadi, bilješke s predavanja, Građevinski fakultet, Zagreb

5) Fresl, K.; Gidak, P.; Vrančić R.: Poopćene minimalne mreže u oblikovanju prednapetih konstrukcija od užadi, Građevinski fakultet , Zagreb

6) Šavor, Z.: Vlačne strukture, prezentacije iz predavanja, Arhitektonski fakultet, Zagreb

7) Jecić, Z.: Oblikovanje visećih konstrukcija pomoću modela, Prostor, 5(1997)1, str. 83-96.

8) Penezić, V.: Majstor perolake arhitekture, Vijenac 287, Matica hrvatska, 2005.

9) Došlić, T.: Newtonova metoda, bilješke s predavanja, Građevinski fakultet , Zagreb

10) Došlić, T.: Gauss-Seidelova metoda, bilješke s predavanja, Građevinski fakultet , Zagreb