VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA STAVEBNÍ
ÚSTAV STAVEBNÍ MECHANIKY

FACULTY OF CIVIL ENGINEERING
INSTITUTE OF STRUCTURAL MECHANICS

PRAVDĚPODOBNOSTNÍ OPTIMALIZACE KONSTRUKCÍ

RELIABILITY-BASED STRUCTURAL OPTIMIZATION

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — Bc. ONDŘEJ SLOWIK
AUTHOR

VEDOUCÍ PRÁCE — prof. Ing. DRAHOMÍR NOVÁK, DrSc.
SUPERVISOR

BRNO 2014

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA STAVEBNÍ

Studijní program	N3607 Stavební inženýrství
Typ studijního programu	Navazující magisterský studijní program s prezenční formou studia
Studijní obor	3608T001 Pozemní stavby
Pracoviště	Ústav stavební mechaniky

ZADÁNÍ DIPLOMOVÉ PRÁCE

Diplomant	Bc. Ondřej Slowik
Název	Pravděpodobnostní optimalizace konstrukcí
Vedoucí diplomové práce	prof. Ing. Drahomír Novák, DrSc.
Datum zadání diplomové práce	31. 3. 2013
Datum odevzdání diplomové práce	17. 1. 2014

V Brně dne 31. 3. 2013

.....	.....
prof. Ing. Drahomír Novák, DrSc. Vedoucí ústavu	prof. Ing. Rostislav Drochytka, CSc., MBA Děkan Fakulty stavební VUT

Podklady a literatura

Sborníky konferencí ICOSSAR a ICASP.
FREET - User´s and Theory Guides, Brno/Červenka Consulting, V 1.5. 2011.
SARA - User´s and Theory Guides, Červenka Consulting, 2010.
ATENA - User´s and Theory Guides, Červenka Consulting, 2010.

Zásady pro vypracování

Téma je zaměřeno na vývoj a testování postupů RBO - reliability-based optimization, kde je třeba zahrnout pravděpodobnostní omezující podmínky. Využívat se bude spolehlivostní software FREET a metoda pseudostochatické optimalizace založená na statistické simulaci LHS. Diplomant vytvoří softwarový nástroj, který bude software FREET využívat. Předpokládá se nejdříve ověření vyvinutých prostředků na jednoduchých optimalizačních případech, a poté aplikace na betonovou konstrukci (např. most) s možností využití nelineárního FEM výpočtu.

Struktura bakalářské/diplomové práce

VŠKP vypracujte a rozdělte podle dále uvedené struktury:

1. Textová část VŠKP zpracovaná podle Směrnice rektora "Úprava, odevzdání, zveřejňování a uchovávání vysokoškolských kvalifikačních prací" a Směrnice děkana "Úprava, odevzdání, zveřejňování a uchovávání vysokoškolských kvalifikačních prací na FAST VUT" (povinná součást VŠKP).
2. Přílohy textové části VŠKP zpracované podle Směrnice rektora "Úprava, odevzdání, zveřejňování a uchovávání vysokoškolských kvalifikačních prací" a Směrnice děkana "Úprava, odevzdání, zveřejňování a uchovávání vysokoškolských kvalifikačních prací na FAST VUT" (nepovinná součást VŠKP v případě, že přílohy nejsou součástí textové části VŠKP, ale textovou část doplňují).

.....

prof. Ing. Drahomír Novák, DrSc.
Vedoucí diplomové práce

ABSTRAKT

Práce demonstruje čtenáři význam optimalizace a pravděpodobnostního posuzování konstrukcí pro problémy stavebního inženýrství. Ve 2. kapitole je blíže popsáno propojení dříve navržených optimalizačních technik s pravděpodobnostním posouzením v podobě omezujících podmínek optimalizace. Pro účely statistického testování a demonstrace účinnosti navrhovaných postupů byl vyvinut akademický software, který navržené algoritmy automatizuje. Kapitola 3. obsahuje přehled výsledků testování autorem navržené optimalizační metody (tzv. Cílené víceúrovňové vzorkování) včetně porovnání s jinými optimalizačními technikami. V závěru práce je demonstrována aplikace popsaných postupů na vybrané optimalizační a spolehlivostní úlohy. V textu představené metody vycházejí z inženýrského přístupu k optimalizačním problémům a kladou si za cíl definovat jednoduše uchopitelný optimalizační algoritmus, jenž by po patřičné automatizaci mohl sloužit účelům běžné inženýrské praxe.

KLÍČOVÁ SLOVA

Optimalizace, Spolehlivostní posouzení, Cílené víceúrovňové vzorkování, LHS, Monte Carlo, Latin Hypercube Sampling, Optimalizační metody, Pravděpodobnost poruchy, Spolehlivostní optimalizace, Simulované žíhání, Genetické algoritmy, Index spolehlivosti, Aproximační metody, Analýza s malým počtem vzorků

ABSTRACT

This thesis presents the reader the importance of optimization and probabilistic assessment of structures for civil engineering problems. Chapter 2 further investigates the combination between previously proposed optimization techniques and probabilistic assessment in the form of optimization constraints. Academic software has been developed for the purposes of demonstrating the effectiveness of the suggested methods and their statistical testing. 3th chapter summarizes the results of testing previously described optimization method (called Aimed Multilevel Sampling), including a comparison with other optimization techniques. In the final part of the thesis, described procedures have been demonstrated on the selected optimization and reliability problems. The methods described in text represents engineering approach to optimization problems and aims to introduce a simple and transparent optimization algorithm, which could serve to the practical engineering purposes.

KEYWORDS

Optimization, Reliability assessment, Aimed Multilevel Sampling, LHS, Monte Carlo, Latin Hypercube Sampling, Optimization methods, Probability of failure, Reliability based design optimization, Simulated annealing, Genetic algorithms, Reliability Index, Approximation methods, Small sample analysis

BIBLIOGRAFICKÁ CITACE

SLOWIK, O. Pravděpodobnostní optimalizace konstrukcí. Brno: Vysoké učení technické v Brně, Fakulta stavební, 2014. 103 s. Vedoucí diplomové práce prof. Ing. Drahomír Novák, DrSc.

PROHLÁŠENÍ

Prohlašuji, že jsem diplomovou práci zpracoval samostatně a že jsem uvedl všechny použité informační zdroje.

V Brně dne 17. Ledna 2014 ……..……..……..

Ondřej Slowik

PODĚKOVÁNÍ

Rád bych poděkoval především prof. Ing. Drahomíru Novákovi, DrSc. za odborné vedení a cenné rady během tvorby této práce. Dále pak Ing. Radoslavu Rusinovi, Ph.D a Ing. Davidu Lehkému Ph.D. za pomoc ve specifických oblastech programování. V neposlední řadě patří můj vděk rodičům a známým, kteří poskytovali potřebné zázemí během celého období mého studia.

OBSAH

ÚVOD

MOTIVACE

Rozvoj konkurenčního prostředí trvající od konce 18. století a z něj vyplývající požadavek na efektivní výrobu kvalitních výrobků ve všech průmyslových odvětvích nutí výrobce investovat nemalé prostředky do optimalizace výrobních procesů, logistiky a výrobků samotných tak, aby ekonomické náklady spojené s výrobou a distribucí byly co možná nejnižší. Ke zmíněnému konkurenčnímu tlaku se navíc pomalu přidává i tlak společenský. Prudkému nárůstu zalidnění v průběhu 20. století je úměrné navýšení spotřeby dostupných přírodních zdrojů, jež je navíc umocněno rostoucími životními nároky obyvatelstva. Efektivní využívání ekonomických zdrojů se stává nezbytným pro pokrytí potřeb světového obyvatelstva. Matematická optimalizace se proto stále více uplatňuje na všech úrovních plánování produkce napříč průmyslovými obory.

Současně s enormním populačním nárůstem a technologickou revolucí 20. století dochází k budování stále složitějších konstrukcí budov a infrastruktury k zajištění potřeb rostoucích městských aglomerací. V důsledku velkého významu těchto staveb vznikla potřeba kvantifikovat jejich spolehlivost a hodnotu rizika spojeného s případným selháním. V průběhu dvacátého století byly vyvinuty tři základní normové metody návrhu staveb dnes běžně užívané po celém světě: metoda dovolených namáhání, metoda stupně bezpečnosti a metoda dílčích součinitelů spolehlivosti [1]. Ačkoli se zmíněné postupy snaží spolehlivost implicitně postihnout, žádná z těchto metod nevypovídá nic o spolehlivosti konstrukce jako celku. V extrémních případech tak stavby navržené dle výše zmíněných metod mohou být nehospodárně předimenzovány či naopak spolehlivostně deficitní. Výkonné počítače dnes umožňují poměrně přesnou kvantifikaci pravděpodobnosti poruchy. Plně pravděpodobnostní přístup k posuzování konstrukcí se také již stal součástí normových předpisů vyspělých zemí. Existuje mnoho variant softwaru umožňujících spolehlivostní modelování. Pokročilé spolehlivostní posuzování konstrukcí je tedy předpokládaným budoucím trendem.

Pomocí vyspělých matematických metod jsme dnes schopni modelovat široké spektrum reálných problémů. Analytické řešení je však možné pouze u malé skupiny těchto úloh za současného přijetí specifických zjednodušení. Komplexní "přesné" řešení složitých modelů je možné díky různým variantám algoritmů převádějících řešení na systémy polynomiálních rovnic (především různé varianty metody konečných prvků). Takové řešení je samo o sobě náročné na výpočetní výkon. Samotné optimalizační problémy se dále dělí dle své složitosti (bez ohledu na výpočetní náročnost optimalizované funkce) na deterministické (tedy řešitelné deterministickými optimalizačními metodami) a nedeterministické polynomiální problémy (řešitelné pravděpodobně pouze heuristicky [2], dále jen NP). Příkladem NP úlohy je dobře známý problém obchodního cestujícího viz [3]. Je zřejmé, že NP úlohy jsou v důsledku nutnosti heuristického řešení výpočetně extrémně náročné i v případě relativně jednoduché formulace optimalizované funkce. I přes rychlý rozvoj výpočetní techniky je optimalizace složitých modelů stále časově náročná a vyčíslení všech

možných variant heuristického řešení není reálné (zvláště u spojitě definovaných problémů). Je proto rozvíjeno mnoho postupů, jež zkracují optimalizační fázi procesu a jsou schopny s určitou pravděpodobností nalézt hledané optimum. Vzhledem k vývoji stále přesnějších a náročnějších matematických modelů existuje předpoklad, že dostupný výpočetní výkon deterministických počítačů nebude ani v dohledné době dostačovat k přesnému řešení složitých optimalizačních úloh. Vývoj stochastických optimalizačních metod je tedy nezbytný.

Metody výpočtu spolehlivosti využívají podobných simulačních postupů a stochastických metod jako optimalizační techniky – jedná se také zpravidla o opakované řešení problému. Mnohé části spolehlivostních výpočtů je dokonce možné formulovat jako optimalizační úlohy (např. výpočet indexu spolehlivosti dle Hasofera a Linda [4], nebo zavedení požadovaných statistických závislostí mezi veličinami). Spojení optimalizace modelu s jeho spolehlivostním posouzením ve formě omezující podmínky optimalizace se proto nabízí. Z důvodu výpočetní náročnosti obou procedur bylo toto spojení v minulosti nemyslitelné, díky rozvoji výpočetní techniky a samotných stochastických, simulačních a aproximačních metod dnes již spojení spolehlivostního posouzení s optimalizačním procesem možné je. S ohledem na rychlé navyšování dostupného výpočetního výkonu a vývoj příslušných softwarových nástrojů implikuje spojení optimalizace se spolehlivostním posouzením velký potenciál využití.

Stavební průmysl má mezi jinými průmyslovými obory zvláštní postavení. Téměř všechny oblasti ekonomické činnosti jsou na stavebnictví do jisté míry závislé. Průběh stavebního procesu je ovlivňován okolním prostředím stavby, ať už se jedná o klimatické, ekologické, geologické nebo sociálně ekonomické faktory. Jejich vzájemné spolupůsobení znemožňuje pouhou replikaci dříve realizovaných konstrukcí a vyžaduje individuální přístup ke každému novému projektu. Budovy a inženýrské konstrukce jsou navíc největšími strukturami, které člověk vytváří. Objemy finančních prostředků, prací a materiálu jsou proto ve stavebnictví vyšší než v kterémkoli jiném průmyslovém oboru. S ohledem na každodenní užívání stavebních konstrukcí velkým počtem osob je bezpečnost a spolehlivost staveb základním aspektem každého návrhu. Pokročilé softwarově automatizované prostředky umožňují komplexní modelování konstrukcí metodou konečných prvků. Výpočetní čas potřebný pro přesnou nelineární analýzu konstrukcí je značný, což znemožňuje pracovat s velkým počtem simulací, jež vyžadují klasické optimalizační techniky. Zavedení plně pravděpodobnostního přístupu navrhování pak výpočetní náročnost dále navyšuje. Pro praktickou optimalizaci vstupních hodnot stavebního návrhu je nutné vyvíjet a zdokonalovat optimalizační postupy pracující s co nejnižším počtem simulací (tzv. small sample analysis, dále jen SSA). Vzhledem ke složitosti optimalizovaných funkcí není k dispozici dostatek informací o průběhu funkce potřebných pro účinné (ve smyslu SSA) nastavení vlastností klasických vyhledávacích algoritmů (např. metoda Simulovaného žíhání). Nově vyvíjené techniky by tak měly být co nejobecnější, snadno použitelné a měly by poskytovat nikoli přesné, ale dostatečně kvalitní výsledky optimalizace pro široký sortiment funkcí bez nutnosti

dalších informací pro efektivní nastavení vyhledávacího procesu. Představení algoritmu, jež může uvedenými vlastnostmi disponovat, je proto cílem této práce.

OBECNÝ PŘEHLED V OBLASTI SPOLEHLIVOSTNÍ OPTIMALIZACE

Během druhé poloviny dvacátého století došlo v oblastech optimalizace konstrukcí a spolehlivostní analýzy k významnému pokroku. Deterministická optimalizace návrhů byla součástí plánování již v dřívějších letech napříč všemi průmyslovými obory. Ke snaze minimalizovat náklady a dosáhnout optimálního návrhu přibývá s rozvojem průmyslu a technologie potřeba vyčíslit také bezpečnost, spolehlivost a riziko selhání navrhovaných struktur. Deterministická analýza spolehlivosti obvykle složitých konstrukcí nevystihuje přirozeně přítomné nejistoty každého návrhu, ať už se jedná o statisticky reprezentovanou proměnlivost materiálové kvality, geometrie či dalších vstupních veličin. Tyto nejistoty mohou vzhledem ke značnému množství materiálu potřebného při budování rozměrných a složitých stavebních konstrukcí hrát zásadní roli při vyhodnocení spolehlivosti. Vznikl tedy požadavek na vývoj a využití optimalizačních metod ve spojení se spolehlivostní analýzou konstrukcí pro potřeby vyhodnocení ideálního poměru mezi ekonomickou náročností a bezpečností navrhovaných konstrukcí. Tento fakt poprvé zdůrazňuje A. M. Freudenthal ve druhé polovině 50. let (např. v [5], [6], [7], [8]).

Složitost procesu spolehlivostní optimalizace je silně závislá na množství a typu návrhových proměnných (rozměrech, tvaru, materiálových parametrech atd.). Většina studií prováděných v oblasti spolehlivostní optimalizace do první poloviny 90. let se zaměřovala na optimalizaci tvaru a velikosti. Pouze minimum prací zohledňovalo také možnost optimalizace vstupních parametrů užitých materiálů. V souladu s rozvojem informatiky a výpočetních metod se aplikace zahrnující také optimalizaci materiálových parametrů dnes již objevují častěji. Příkladů aplikujících spolehlivostní optimalizaci na reálné stavební konstrukce je však stále jen minimum [9].

Algoritmizace spolehlivostní optimalizace přináší otázku významu a typu nalezeného optima. Dostupná literatura nabízí několik variant formulace cílové funkce a optimalizačního procesu, např. jako minimalizace celkové očekávané ceny konstrukce včetně nákladů spojených s rizikem kolapsu, minimalizace teoretické pravděpodobnosti poruchy nebo minimalizace hmotnosti konstrukce s ohledem na spolehlivostní omezující podmínky (např. [10]). Mnohdy jsou optimalizační problémy formulovány jako multikriteriální s několika cílovými funkcemi, např. minimalizace očekávané celkové ceny konstrukce se současným požadavkem maximalizace celkové systémové spolehlivosti (např. [11]). Výsledky multikriteriálního optimalizačního procesu jsou pak podkladem pro rozhodování o výsledné podobě návrhu.

Během posuzování konstrukce se spolehlivost vyhodnocuje na různých úrovních návrhu. Celkové selhání konstrukce pod vlivem extrémních zatížení i částečné poškození vlivem selhání jednotlivých konstrukčních částí musí být postiženy ve formulaci spolehlivostní optimalizace. Výsledný návrh musí také respektovat podmínky mezního stavu

trvanlivosti, jež mohou být rozhodujícím faktorem ovlivňujícím požadovanou úroveň spolehlivosti [12]. Ve většině formulací, jež se v literatuře objevují, není časová závislost chování konstrukčních systémů zohledněna. Systémové parametry se však obvykle během životnosti konstrukce mění a zanedbáním časové závislosti konstrukční odezvy se můžeme dopustit hrubé chyby. Z těchto důvodů je nutné navýšení počáteční hladiny spolehlivosti na úroveň, která zajistí, že minimální požadovaná spolehlivost bude dodržena během celé plánované životnosti stavby. Jinou variantou řešení pak může být stanovení rozvrhu údržby a oprav, jež zajistí navrácení na požadovanou hladinu spolehlivosti. Také problematika zajištění potřebné úrovně spolehlivosti během životnosti konstrukce může být implementována do optimalizačního procesu (např. [13], [14]).

Výpočetní náročnost spolehlivostní optimalizace podněcuje k vývoji efektivních metod jak pro výpočet samotné funkční hodnoty cílové funkce, tak pro optimalizační a spolehlivostní výpočetní procedury.

Pro kvantifikaci odezvy konstrukce se dnes běžně využívá metoda konečných prvků (např. [15]). Bližší studium široké problematiky metody konečných prvků, však není cílem tohoto textu.

K vyhodnocení spolehlivosti je nutné vyčíslit pravděpodobnost poruchy, jež je ve své obecné formě dána integrálem (40) (viz kapitola 1.5.2). S ohledem na složitost funkce sdružené hustoty pravděpodobnosti a tvar integrační oblasti je přímá integrace v obecném případě nereálná. Byla proto vyvinuta sada přibližných metod k určení pravděpodobnosti poruchy. Zmíněné metody lze rozdělit na metody simulační a aproximační. Simulační metody jsou založeny na integraci numerickou simulací typu Monte Carlo, např. [16] (viz kapitola 1.3). Nároky na výpočetní čas lze snížit použitím speciálních technik redukce rozptylu viz [17]. Z těchto technik je nejčastěji využívána metoda Importance Sampling, jež poskytuje poměrně přesné odhady pravděpodobnosti poruchy. Tato metoda se řadí mezi tzv. zdokonalené simulační metody spolu s dnes dále rozvíjenými metodami Asymptotic sampling, Adaptive Sampling a Directional Sampling, blíže ve [18], [19], [20] a [21]. Mezi další simulační metody snižující rozptyl odhadu spolehlivostních charakteristik patří např. metoda Conditional [22], či metody Latin Hypercube sampling [9], [23], [24]. Tyto metody přispívají ke snížení statistické chyby odhadu pravděpodobnosti poruchy na minimu a činí tak integraci Monte Carlo velice atraktivní. Aproximační metody buď nahrazují funkci poruchy jednoduchou aproximační funkcí (metody FORM a SORM) a s ní dále pracují pomocí výše zmíněných zdokonalených simulačních metod, např. [25] nebo se aproximuje až empirická distribuční funkce rezervy spolehlivosti vhodným teoretickým modelem (metody Response surface) [26], [27]. V rámci této práce se s ohledem na provedenou algoritmizaci optimalizačního procesu s využitím stávajícího softwaru FReET pracuje výhradně s metodou FORM (viz kapitola 1.6.2), v podobě v jaké je aplikována v rámci zmíněného softwaru [28].

Pokračuje také vývoj účinných optimalizačních algoritmů a metod analýzy sensitivity potřebných pro efektivní spolehlivostní optimalizaci. Mezi takovéto nástroje patří např. Metoda Simulovaného žíhání, [29], [3], [30], či metody Genetických algoritmů, jež nacházejí

inspiraci v procesech přírodního výběru [31], [32]. Bližší informace o genetických algoritmech jsou k dispozici také v [33] [34] a [35]. Existují mnohé další optimalizační postupy, jež jsou však v porovnání se zmíněnými metodami uplatňovány jen okrajově [9]. Popisované optimalizační metody zastupují pouze metody stochastické, popis deterministických optimalizačních metod, jako jsou Newtonovy metody či metody gradientní, je mimo rámec této práce.

Pro integraci výše zmíněných numerických metod v rámci uceleného spolehlivostně orientovaného optimalizačního procesu byly vyvinuty různé softwarové prostředky [36], [37]. Novější přístup prezentují např. Pospíšilová, Myšáková a Lepš v [38]. Většina příkladů spolehlivostní optimalizace však zatím pochází hlavně z aplikací v rámci leteckého průmyslu [39].

Spíše okrajové uplatnění spolehlivostní optimalizace ve stavebním průmyslu je dáno konzervativním přístupem stavebních inženýrů a nedostatečnou algoritmizací celého procesu. Nároky na inženýrský čas a potřebná vysoká úroveň znalostí zatím zastiňují benefity spolehlivostní optimalizace. Vývoj několika posledních let však ukazuje, že spolehlivostní optimalizace konstrukcí má své opodstatnění a dá se předpokládat, že zaujme významné místo při navrhování konstrukcí.

1. TEORETICKÝ ZÁKLAD

1.1 DETERMINISTICKÁ FORMULACE OPTIMALIZAČNÍHO PROBLÉMU

Cílem optimalizace je nalezení extrému (tedy maxima či minima) dané funkce. Cílová funkce optimalizace je formulována na základě požadavku srovnání různých variant návrhu v rámci vyšetřovaného systému. U většiny zkoumaných problémů je nutné vymezení hranic přípustné optimalizace, např. stanovením požadavku, aby geometrické parametry byly kladné a v rámci reálných mezí. Tyto požadavky jsou vyjádřeny v podobě omezujících podmínek formulovaných pomocí hraničních rovností a / nebo nerovností, zahrnutých v matematickém modelu optimalizace. Deterministická definice optimalizačního problému pak může být vyjádřena následovně:

Najděme takový vektor proměnných $\boldsymbol{x} = (x_1, x_2, ..., x_n) \in R^n$, pro nějž platí, že funkční hodnota zkoumané cílové funkce definované na oblasti $S \in R^n$ je extrémní. Pro případ minimalizace tedy:

$$f(\boldsymbol{x}) = f(x_1, x_2, \dots., x_n) \rightarrow min \tag{1}$$

Za předpokladu existence omezujících podmínek v podobě hraničních rovností a nerovností:

$$h_j(\boldsymbol{x}) \equiv h_j(x_1, x_2, \dots., x_n) = 0 \qquad j = 1 \text{ až } p \tag{2}$$

$$g_i(\boldsymbol{x}) \equiv g_i(x_1, x_2, \dots., x_n) \leq 0 \qquad i = 1 \text{ až } m \tag{3}$$

Vztahy (2) a (3) definují hranice přípustné optimalizace na základě vstupních návrhových požadavků. Hodnoty m a p vyjadřují počet každého z typů omezujících podmínek. Hodnota n představuje počet dimenzí optimalizačního problému.

V závislosti na typu cílové funkce a omezujících podmínkách byla vyvinuta řada různých algoritmů matematického programování, např. [40]. Deterministické optimalizační metody užívané ve zmíněných algoritmech lze dělit dle druhu informace, s níž pracují na:

- metody vyhledávací, vyžadující pouze hodnoty cílové funkce $f(\boldsymbol{x})$,
- metody gradientní, využívající první parciální derivace $f(\boldsymbol{x})$ (tzv. strategie prvního řádu),
- Newtonovy metody, využívající také druhé parciální derivace $f(\boldsymbol{x})$ (tzv. strategie druhého řádu).

Nevýhodou uvedených deterministických metod je, že žádná z nich nevede obecně k nalezení globálního extrému zkoumané funkce. Většina deterministických metod se také potýká s problémy v případě nespojitosti cílové funkce. Podrobnější popis těchto metod je k dispozici například v [41].

Problém spolehlivostní optimalizace může být definován podobně. Rozdíl je v definici proměnných vstupujících do výpočtu, jež jsou ve stochastické definici optimalizačního problému se spolehlivostními omezujícími podmínkami uvažovány jako náhodné.

1.2 PRAVDĚPODOBNOSTNÍ FORMULACE OPTIMALIZAČNÍHO PROBLÉMU

Základním předpokladem spolehlivostní optimalizace je modelování jak zatížení, tak konstrukční odezvy pomocí náhodných proměnných. V závislosti na požadované robustnosti a výstižnosti matematického modelu je tedy nutná randomizace některých jeho vstupních parametrů. Jsou-li některé z funkčních parametrů uvažovány jako náhodné, pak je samotná analyzovaná funkce také funkcí náhodnou. Obecnou stochastickou formulaci problému spolehlivostní optimalizace potom můžeme vyjádřit například takto:

$$f\left(\boldsymbol{x}, \boldsymbol{Y}, \boldsymbol{r}(\boldsymbol{x}, \boldsymbol{Y}, \boldsymbol{y}')\right) \rightarrow min \tag{4}$$

se zohledněním omezujících podmínek:

$$h_j\left(\boldsymbol{x}, \boldsymbol{Y}, \boldsymbol{r}(\boldsymbol{x}, \boldsymbol{Y}, \boldsymbol{y}')\right) = 0 \qquad j = 1 \text{ až } p \tag{5}$$

$$g_i\left(\boldsymbol{x}, \boldsymbol{Y}, \boldsymbol{r}(\boldsymbol{x}, \boldsymbol{Y}, \boldsymbol{y}')\right) \leq 0 \qquad i = 1 \text{ až } m \tag{6}$$

kde $\boldsymbol{x}$ je vektor deterministických návrhových proměnných, $\boldsymbol{Y}$ je vektor náhodných proměnných, $\boldsymbol{r}$ je vektor uvažovaných pravděpodobnostních funkcí a $\boldsymbol{y}'$ jsou statistické parametry náhodných proměnných. Čísla p a m pak vyjadřují počty omezujících podmínek.

V souvislosti se současnou aplikací spolehlivostního vyhodnocení a stochastických optimalizačních metod v rámci jedné procedury je třeba podotknout, že vektor $\boldsymbol{y}'$ může zahrnovat dvě sady statistických parametrů náhodných proměnných. První sada statistických parametrů reprezentuje randomizaci náhodných proměnných vystihující přirozené chování statistických veličin vyhodnocené na základě experimentu. Tato sada statistických parametrů je pak využita pro spolehlivostní výpočty. Druhá sada statistických parametrů náhodných veličin je pak použita pro účely optimalizace. Pro optimalizaci se tedy znáhodňují ty parametry, jejichž optimální vstupní kombinaci hledáme. Statistické parametry se pak volí s ohledem na volbu optimalizační metody tak, aby byl návrhový prostor pokryt co možná nejrovnoměrněji.

Obecně je návrh konstrukce závislý na veličinách kvantifikujících odezvu zkoumané konstrukce na působící zatížení (např. přetvoření a napětí). Stanovme proto funkci odezvy konstrukce jako:

$$Y_i = Y_i(\boldsymbol{A}(\omega), \boldsymbol{x}) \qquad i = 1 \text{ až } m \tag{7}$$

kde $\boldsymbol{x}$ je návrhový vektor deterministických parametrů a $\boldsymbol{A}(\omega)$ je vektor náhodných parametrů vyšetřované konstrukce (např. zatížení či pevnosti). Konstrukční požadavky pak mohou být formulovány jako:

$$y_{il} \leq Y_i(\boldsymbol{A}(\omega), \boldsymbol{x}) \leq y_{iu} \qquad i = 1 \text{ až } m \tag{8}$$

s danými hranicemi y_{il} a y_{iu}. Omezující podmínky pro deterministické návrhové proměnné lze stanovit jako:

$$x_{il} \leq x_i \leq x_{iu} \qquad i = 1 \text{ až } n \tag{9}$$

Spolehlivostní omezující podmínky lze vyjádřit pomocí pravděpodobnostní funkce:

$$P_i(\boldsymbol{x}) = P(y_{il} \leq Y_i(\boldsymbol{A}(\omega), \boldsymbol{x}) \leq y_{iu}) \qquad i = 1 \text{ až } m \tag{10}$$

Zaveďme nyní funkci celkové ceny konstrukce $c=c(\boldsymbol{z})$, jež bude sloužit jako hlavní kritérium vhodnosti návrhu. Optimální návrhový vektor vstupních hodnot $\boldsymbol{z}^*$ složený z vektoru deterministických návrhových proměnných $\boldsymbol{x}$ a vektoru náhodných proměnných $\boldsymbol{A}(\omega)$ je určen pomocí stochastické optimalizace (např. [42]). Formulace optimalizačního problému pak může být pojata jako maximalizace spolehlivosti, za předpokladu existence omezující podmínky definované jako maximální přípustná cena konstrukce.

$$P(\boldsymbol{x}) = P((y_{il} \leq Y_i(\boldsymbol{A}(\omega), \boldsymbol{x}) \leq y_{iu}, i = 1 \text{ až } m) \rightarrow max \tag{11}$$

omezeno podmínkami:

$$c(\boldsymbol{z}) \leq c_{max} \tag{12}$$

$$x_{il} \leq x_i \leq x_{iu} \qquad i = 1 \text{ až } n \tag{13}$$

kde návrhový prostor pro výpočet pravděpodobnosti je definován jako:

$$(\Omega, \Sigma, P), \omega \in \Omega \tag{14}$$

s daným rozdělením pravděpodobnosti. Ω je zde vzorkovací prostor pro výpočet pravděpodobnosti, Σ je pak kompletní návrhový prostor veličin.

Z uvedené formulace je zřejmá výpočetní náročnost procesu spolehlivostní optimalizace. Pro účely stochastické optimalizace je nutné opakovaně generovat náhodné realizace v rámci návrhového prostoru Σ. Pro každou z těchto realizací je dále třeba vypočíst spolehlivost (v obecném případě náročnou integrací vztahu (40)). Vyčíslení spolehlivosti je přitom spjato s opakovaným vyhodnocováním odezvy konstrukce, což může při použití složitého modelu k určení odezvy přinášet enormní nároky na výpočetní čas. Vznikla proto celá řada aproximačních metod, jež si kladou za cíl snížit výpočetní náročnost

spolehlivostního posouzení (FORM, SORM, Response surface methods) [25], [26], [27], a stejně tak i optimalizačních technik pro analýzu s malým počtem vzorků [9], [35].

1.3 METODY GENEROVÁNÍ REALIZACÍ V RÁMCI NÁVRHOVÉHO PROSTORU

V postupech spolehlivostní optimalizace mají výsadní postavení metody umožňující generování náhodných realizací v rámci návrhového prostoru. Simulační metody jsou základním nástrojem jak pro spolehlivostní posouzení, tak pro optimalizační proces. Jejich užití můžeme registrovat v mnoha dalších odvětvích numerické analýzy.

Pravděpodobně první popsaný případ aplikace stochastické simulační metody na řešení matematického problému se datuje do roku 1777 (tzv. problém Buffonovy jehly). Francouzský matematik Georges-Louis Leclerc Comte de Buffon se pokoušel odhadnout hodnotu Ludolfova čísla π náhodným vrháním jehly na linkovaný papír. Pravděpodobnost jevu, kdy jehla délky rovné vzdálenosti mezi linkami po dopadu na papír zůstane ležet na papíře tak, že protíná některou z linek, je rovna $p = \frac{2}{\pi}$. Po statistickém vyhodnocení počtu úspěšných a neúspěšných realizací tohoto náhodného jevu bylo možné určit přibližně hodnotu čísla π. Přesnost takto získané hodnoty narůstá s rostoucím počtem realizací náhodného jevu.

Tato metoda byla poprvé systematicky použita kolem roku 1930 Enrico Fermim při generování náhodných čísel k výpočtu vlastností neutronu. Později byla pojmenována Stanislawem Ulamem Monte Carlo [9]. Na základě metody Monte Carlo byly postupně odvozeny další simulační metody. Na konci 70. let se objevila metoda Latin Hypercube Sampling. V průběhu 80. let byly vyvinuty tzv. zdokonalené simulační metody sloužící k redukci výpočetních nároků spolehlivostní analýzy. Mezi tyto metody patří např. Importance Sampling, Adaptive Sampling, Directional Sampling nebo Asymptotic Sampling. Popisu vybraných simulačních metod se věnují následující kapitoly.

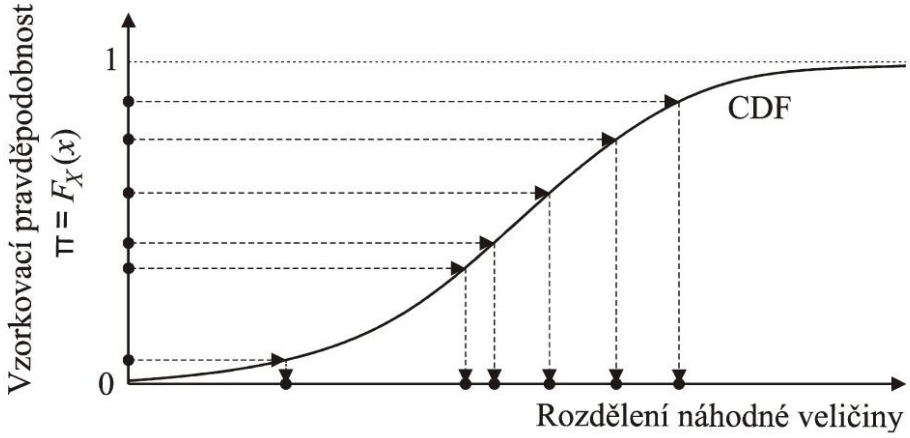
1.3.1 KLASICKÁ METODA MONTE CARLO

Pro účely numerické simulace pomocí některého z typů metody Monte Carlo je nutné nadefinovat matematický model problému a zajistit generování proměnných hodnot náhodných veličin v závislosti na jejich teoretickém modelu rozdělení pravděpodobnosti. K tomu je nezbytné užití generátoru náhodných čísel rovnoměrně rozdělených na intervalu (0;1) (generátory náhodných čísel na intervalu 0 až 1 jsou dnes standardně součástí mnoha programovacích jazyků, blíže v [1],[17]). Na základě vygenerovaného pseudonáhodného čísla $u_{i,j}$ se generuje realizace náhodné veličiny s daným pravděpodobnostním rozdělením tak, že se hledá realizace x_i v j-té simulaci, pro niž platí, že funkční hodnota distribuční funkce jejího pravděpodobnostního rozdělení v x_i je rovna vygenerovanému pseudonáhodnému číslu $u_{i,j}$ (viz Obr. 1 a vztah (15)). Vygenerovaná čísla nejsou zcela náhodná, neboť proces generování lze kdykoliv zopakovat se stejnými vstupními podmínkami a řada vygenerovaných čísel se bude s určitou periodou opakovat, hodnota této periody musí být tudíž vysoká. [9], [1]

$$x_{i,j} = \Phi_{X_i}^{-1}(u_{i,j}) \tag{15}$$

kde $\Phi_{X_i}^{-1}(u_{i,j})$ je inverzní distribuční funkce náhodné veličiny X_i.

Na základě takto vygenerovaných náhodných vektorů X $(x_1;x_2;...;x_n)$ se vypočtou funkční hodnoty $f_{(X)}$. Soubor získaných funkčních hodnot je dále zpracován metodami matematické statistiky.



Obr. 1 Princip výběru realizací náhodných veličin metodou Monte Carlo - převzato ze [43]

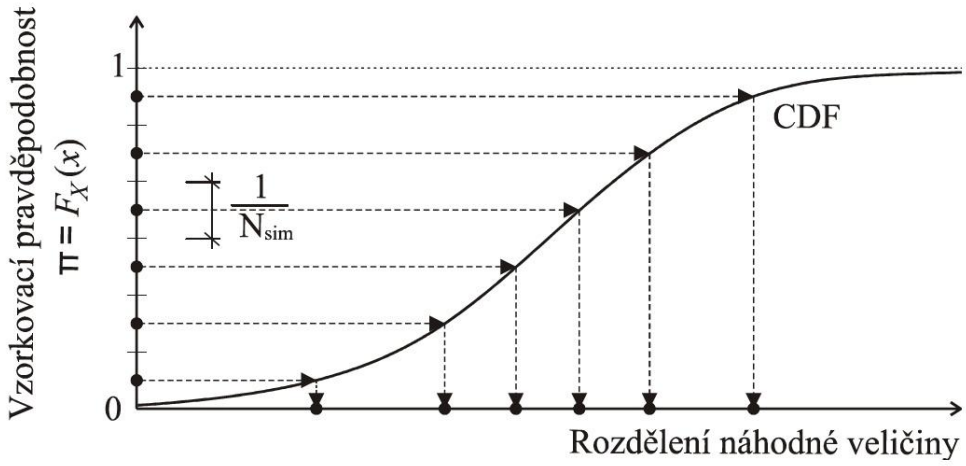
Výhodou těchto metod je jejich transparentnost a srozumitelnost. Nevýhodou pak může být vysoký počet iterací, jež je činí obtížně použitelnými pro výpočetně náročné problémy spolehlivostní optimalizace. Hlavním nedostatkem metody Monte Carlo je nerovnoměrné pokrytí návrhového prostoru při SSA. Jsou-li počty vygenerovaných hodnot nízké, pak histogramy simulací zpravidla neodpovídají funkcím hustot pravděpodobnosti, jež byly přiděleny jednotlivým proměnným. Tuto nevýhodu odstraňují metody typu Latin Hypercube Sampling (LHS) popsané v dalším textu.

1.3.2 METODY LHS (LATIN HYPERCUBE SAMPLING)

LHS jsou speciálními typy simulací Monte Carlo využívající pravidelné rozvrstvení oboru hodnot pravděpodobnostní distribuční funkce náhodné proměnné na intervaly s ekvivalentní pravděpodobností *1/Nsim*, kde *Nsim* je počet realizací náhodné veličiny (viz. Obr. 2). Tyto metody jsou kategorizovány také jako metody redukce rozptylu statistik. Z každého sub intervalu spojité distribuční funkce $(j = 1, ..., N_{sim})$ je vybrána právě jedna realizace náhodné proměnné $x_{i,j}$. LHS zajišťuje rovnoměrné pokrytí prostoru s mnoha náhodnými proměnnými pouze s minimálním počtem vzorků. V závislosti na způsobu výběru hodnoty $x_{i,j}$ ze sub intervalu rozlišujeme několik typů LHS:

- Lattice sampling by Patterson (LHS median)
- LHS-random

- LHS mean



Obr. 2 Princip výběru realizací náhodných veličin metodou LHS – přrvzato ze [43]

LATTICE SAMPLING BY PATTERSON (LHS MEDIAN)

Realizace náhodné proměnné $x_{i,j}$ je v této variantě LHS generována jako:

$$x_{i,j} = F_i^{-1}\left(\frac{\pi_i(j) - 0{,}5}{N_{sim}}\right) \tag{16}$$

Kde $\pi_i(1),\ldots, \pi_i(N_{sim})$ je náhodná permutace z $1,\ldots, N_{sim}$. F_i^{-1} je inverzní distribuční funkce této náhodné proměnné.

Nevýhodou centrické verze LHS je soustředění vzorků na středy sub intervalů, to vede k redukci variability vzorků z okrajových částí (tails) funkce hustoty pravděpodobnosti (dále jen PDF- probability density function), které nejvíce ovlivňují šikmost a špičatost rozdělení veličin.

LHS – RANDOM

Tato metoda je téměř analogická s metodou (16) s tím rozdílem, že reprezentant není vybírán jako střed sub intervalu, ale může být vybrána jakákoli realizace náhodné veličiny s funkční hodnotou distribuční funkce v mezích sub intervalu. Toto se projeví na vztahu (16) výměnou hodnoty 0,5 za hodnotu U_j^i (náhodně vygenerované číslo z intervalu 0 až 1).

LHS MEAN

Tato metoda odstraňuje problém variability vzorků z okrajových částí PDF výběrem vzorku odpovídajících střední hodnotě sub intervalu PDF.

Realizace náhodných veličin jsou tak generovány podle vztahu:

$$x_{i,j} = \frac{\int_{\xi_{i,j-1}}^{\xi_{i,j}} x f_i(X) dx}{\int_{\xi_{i,j-1}}^{\xi_{i,j}} f_i(X)\, dx} = N_{sim} \int_{\xi_{i,j-1}}^{\xi_{i,j}} x f_i(X) dx \quad (17)$$

$f_i(X)$ je funkce hustoty pravděpodobnosti neznámé X_i. Hranice integrační oblasti $\xi_{i,j} = F_i^{-1}(j/N_{sim})$ pro $j=1,\ldots,N_{sim}$

Tuto simulační metodu lze s výhodou používat u proměnných s funkcí hustoty pravděpodobnosti, jež je snadno integrovatelná. V případech, kdy je analytické řešení náročné nebo nemožné, je nutné přistoupit k numerické integraci (tím vzroste výpočetní náročnost mnohdy nad únosnou mez). Vzorky vybírané dle vztahů (16) a (17) jsou téměř identické, vyjma reprezentantů sub intervalů na krajích PDF.

Proti klasické metodě Monte Carlo poskytují metody LHS uspokojivou přesnost i při malém počtu simulací. Jsou proto vhodné pro optimalizaci výpočetně náročných problémů s malým množstvím provedených simulací (SSA) běžných ve stavební praxi [9]. Drobnou nevýhodou metod typu LHS je nemožnost přidávání dodatečných simulací v průběhu procesu. Tuto nevýhodu se snaží odstranit metoda Hierarchical Subset Latin Hypercube Sampling (HSLHS), jež umožňuje hierarchicky přidávat vždy dvojnásobný počet simulací oproti předchozí úrovni vzorkování, blíže např. v [43]. S ohledem na generování vzorků v rámci každého sub-intervalu distribuční funkce náhodné veličiny odpovídá histogram generovaných vzorků definované funkci hustoty pravděpodobnosti i v případě malého množství simulací.

Po vygenerování jednotlivých realizací všech náhodných veličin $x_1;x_2;\ldots;x_n$ je nutné sestavit náhodné vektory X_j $(x_1;x_2;\ldots;x_n)$ pro $(j = 1, \ldots, N_{sim})$. K tomuto účelu je vhodné definovat tzv. tabulku náhodných permutací, jež obsahuje náhodné permutace čísel $1, 2, \ldots, N_{sim}$ pro každou náhodnou veličinu (Tab. 1).

Tab. 1 Tabulka náhodných permutací

N_{sim}	$x_1;x_2;\ldots;x_n$ n=6					
	1	2	3	4	5	6
1	2	8	4	2	8	10
2	5	4	1	9	9	4
3	9	10	8	3	1	5
4	7	1	5	4	7	8
5	8	9	2	6	10	3
6	3	7	9	5	6	9
7	6	5	6	10	3	2
8	4	6	3	8	5	7
9	1	3	10	1	4	6
10	10	2	7	7	2	1

Počet řádků sestavené tabulky odpovídá N_{sim} . Počet sloupců tabulky je roven počtu náhodných veličin. Generované realizace náhodných veličin jsou pak v každém sloupci seřazeny dle náhodné permutace čísel *1, 2, ..., N_{sim}*. Jednotlivé náhodné vektory X_j pak odpovídají řádkům tabulky [1].

Korelaci mezi vstupními veličinami lze po zadání požadované korelační matice zavést iterativní úpravou permutací sloupců vzorků náhodných veličin v rámci tabulky. Hledá se taková sada sloupcových permutací, pro niž platí, že rozdíl mezi požadovanou a získanou korelační maticí je minimální. Jedná se tedy o optimalizační problém řešitelný některým z optimalizačních algoritmů (např. v softwaru FReET je pro zavedení korelace využita metoda Simulovaného žíhání). Jako míru statistické závislosti dvou souborů je v obecném případě vhodné použít Spearmanův koeficient pořadové korelace daný vztahem:

$$r^s = 1 - \frac{6\sum_{i=1}^{n} d_i^2}{n(n-1)(n+1)} \tag{18}$$

1.3.3 DALŠÍ SIMULAČNÍ METODY

Další část textu je věnována tzv. zdokonaleným simulačním metodám, jež jsou vyvíjeny od počátku 80. let především pro potřeby kvantifikace pravděpodobnosti poruchy. Zmíněné metody bývají v literatuře též označovány jako metody redukce rozptylu v okolí funkce mezního stavu a výrazně snižují počet simulací nutný k vyčíslení malých hodnot pravděpodobnosti poruchy.

IMPORTANCE SAMPLING

Metody typu Importance sampling jsou účinným nástrojem pro výpočet malých pravděpodobností. Oproti klasické Metodě Monte Carlo výrazně snižují počet simulací nutných ke spolehlivému odhadu pravděpodobnosti. Jejich využití je většinou spojeno s aproximačními metodami typu FORM nebo SORM, jedná se tedy o metody využívané především ve spolehlivostních postupech 2. úrovně [1]. Podrobný popis metod typu Importance sampling je nad rámec tohoto textu a nalezneme jej např. v [18], [19], [20] a [21]. Zaměřme se proto pouze na výklad základní podstaty těchto metod.

Základní myšlenkou metody je generování náhodných veličin ne podle jejich skutečné hustoty rozdělení pravděpodobnosti $f_x(\boldsymbol{X})$, ale podle tzv. váhové funkce $h_y(\boldsymbol{X})$, která představuje uměle zvolené rozdělení pravděpodobnosti nastavené tak, aby k poruše docházelo velmi často. Simulace je tedy zaměřena především okolo oblasti poruchy D_f.

Obr. 3 zachycuje dvourozměrný případ návrhového prostoru. Je zde vyobrazena funkce poruchy $g(\boldsymbol{X})$, rozdělující návrhový prostor na oblast bezpečnou $[g(\boldsymbol{X}) \geq 0]$ a oblast poruchy $[g(\boldsymbol{X}) < 0]$, dále pak původní funkce hustoty rozdělení pravděpodobnosti $f_x(\boldsymbol{X})$ a váhová funkce $h_y(\boldsymbol{X})$. Bod ležící na hranici $g(\boldsymbol{X}) = 0$ s minimální vzdáleností od počátku v normalizovaném prostoru je tzv. návrhový bod (souřadnice u_1^*, u_2^*), který lze získat

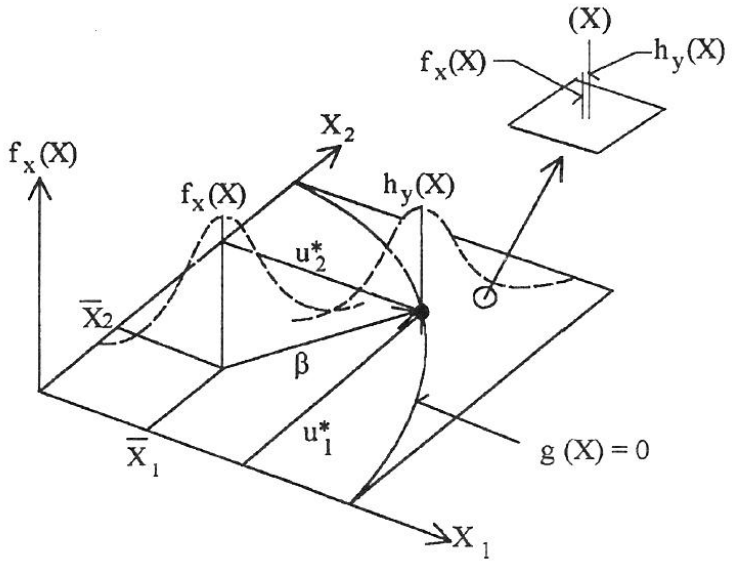
řešením optimalizační úlohy vyjádřené vztahy v normovaném prostoru náhodných veličin $\boldsymbol{Y}$ [4]:

$$\beta = \sqrt{\boldsymbol{Y}^T\boldsymbol{Y}} \rightarrow min \tag{19}$$

Za podmínky:

$$g(\boldsymbol{X}) = 0 \tag{20}$$

kde β je vzdálenost návrhového bodu od počátku. Minimální hodnota β je pak známá jako Hasofer Lindův index spolehlivosti [4].



Obr. 3 Importance sampling s použitím návrhového bodu ve 2D – převzato z [1]

Vlivem umělého zavedení váhové funkce $h_y(\boldsymbol{X})$ probíhá simulace v poněkud jiném prostoru náhodných veličin než je tomu v případě klasické simulace Monte Carlo. Tuto změnu je nutné kompenzovat v integrálním výrazu pro pravděpodobnost poruchy (47). Vztah tedy rozšíříme zavedením váhové funkce na tvar:

$$p_f = \int_{\Omega_x} 1[g(X) < 0] \frac{f_x(\boldsymbol{X})}{h_y(\boldsymbol{X})} \, h_y(\boldsymbol{X}) d\boldsymbol{X} \tag{21}$$

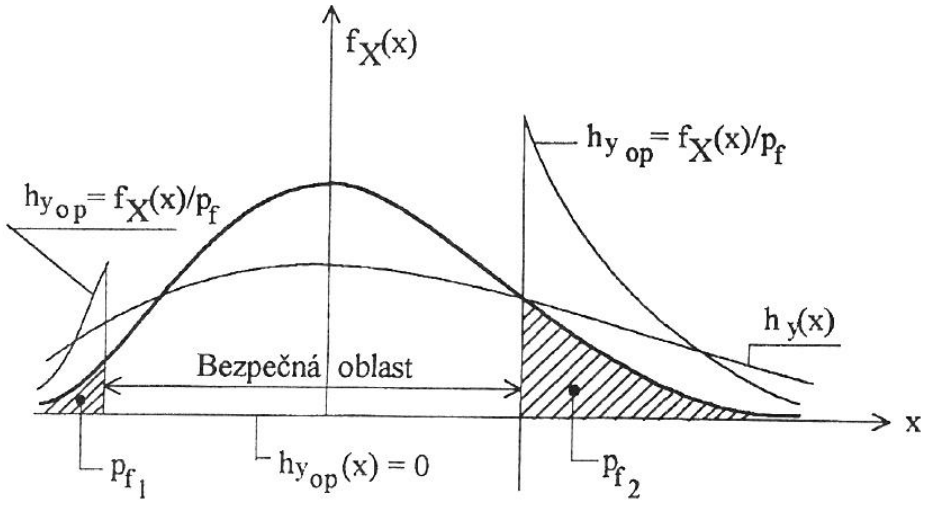
kde $1[g(X) < 0] = 1$ pro $\boldsymbol{X}$ patřící do oblasti poruchy a $1[g(X) < 0] = 0$ pro $\boldsymbol{X}$ mimo oblast poruchy.

Simulace tedy neprobíhá v oblasti Ω_x (což by vedlo k vykrácení váhové funkce ve vztahu (21) a rovnice by reprezentovala pouze klasickou simulaci Monte Carlo), ale je realizována v oblasti Ω_y se zohledněním váhové funkce. Náhrada integrálu (47) za sumaci potom odpovídá vztahu (22)

$$p_f = \frac{1}{N}\sum_{i=1}^{N} 1[g(X) < 0]\frac{f_x(\boldsymbol{X_i})}{h_y(\boldsymbol{X_i})} \tag{22}$$

kde N je počet simulací. Tímto způsobem lze počítat velmi malé pravděpodobnosti i pro malá N.

V praktických úlohách se často setkáváme s případy, kdy před vlastním výpočtem není dostatek informací o zkoumané oblasti poruchy. V takovém případě není jasné, kolem jakých bodů by se mělo vzorkování zaměřit. Využívá se proto technika umístění postupu "Importance sampling" do středních hodnot náhodných veličin s použitím velkých rozptylů váhových funkcí. Tento typ Importance sampling může být použit pro prakticky jakýkoli spolehlivostní problém [18]. Technika Importace sampling s umístěním váhových funkcí do středních hodnot je pro 1D případ demonstrována na Obr. 4.



Obr. 4 Importance sampling s použitím středních hodnot (1D případ) – převzato z [1]

Je evidentní, že tento případ vyžaduje širokou oblast pro simulaci. Optimální předem neznámé váhové funkce jsou na Obr. 4 označeny jako $h_{y_{op}}$, tyto by pak měly být proporcionální k hodnotě *pf*, více v [18].

ASYMPTOTIC SAMPLING

Asymptotitic sampling je poměrně nová technika popsaná např. v [44] určená k odhadu indexu spolehlivosti na základě asymptotického chování pravděpodobnosti poruchy v n-dimensionálním normovaném prostoru [45]. Základní myšlenkou je navýšení počtu vzorků generovaných v oblasti poruchy pomocí zvětšení směrodatných odchylek náhodných veličin. Index spolehlivosti je pak funkcí faktoru zvětšení směrodatných odchylek $f = \frac{1}{\sigma}$ a je definován vztahem:

$$\frac{\beta}{f} = A + \frac{B}{f^2} \tag{23}$$

Koeficienty A a B jsou získávány nelineární regresní analýzou pomocí tzv. podpůrných bodů. V těchto bodech (representovaných v grafu $\beta/f \; vs \; f$) jsou vyhodnoceny hodnoty indexu spolehlivosti β_i pro danou hodnotu f_i. Vyhodnocení podpůrných bodů probíhá pro postupně klesající hodnotu faktoru f. Na začátku procesu je třeba určit následující parametry:

- Počet vzorků k vyčíslení indexu spolehlivosti podpůrných bodů m
- Minimální počet vzorků realizovaných v oblasti poruchy N_0
- Koeficient redukce f_d pro faktor f
- Požadované množství podpůrných bodů K

Pro aktuální hodnotu faktoru f se provede vzorkování návrhového prostoru a vyhodnocení počtu realizací v rámci oblasti poruchy. Je-li tento počet vyšší než požadovaná hodnota N_0, pak je spočtená hodnota indexu spolehlivosti β_i pro danou hodnotu f_i uložena jako podpůrný bod. V opačném případě dojde k redukci faktoru f. Po dosažení požadovaného množství podpůrných bodů K je proces zastaven. Pomocí nelineární regresní analýzy jsou následně určeny parametry A a B. Index spolehlivosti (pro náhodné proměnné s původními směrodatnými odchylkami) je poté vypočten dosazením hodnot A a B do vztahu (23) pro $f = 1$, tedy jako pouhý součet A a B.

Příklad užití Asymptotic sampling v rámci postupů spolehlivostní optimalizace lze nalézt v [38], [45].

ADAPTIVE SAMPLING

D. Basu v [46] dokazuje, že teoreticky nejlepší simulační model bude takový, jež je závislý na pozorovaných hodnotách samotných [47]. Adaptive sampling je souhrnné pojmenování metod vzorkování, jež využívají rozvrstvení vzorkovacího procesu do jednotlivých fází. Data získaná vyhodnocením vzorků fáze předchozí poté ovlivňují simulaci ve fázi následující. Jedná se o celou řadu často velmi složitých metod, jež mají široké uplatnění například v oblastech informatiky, farmaceutického testování, biologie, stochastické optimalizace atd., bližší informace jsou k dispozici např. v [48].

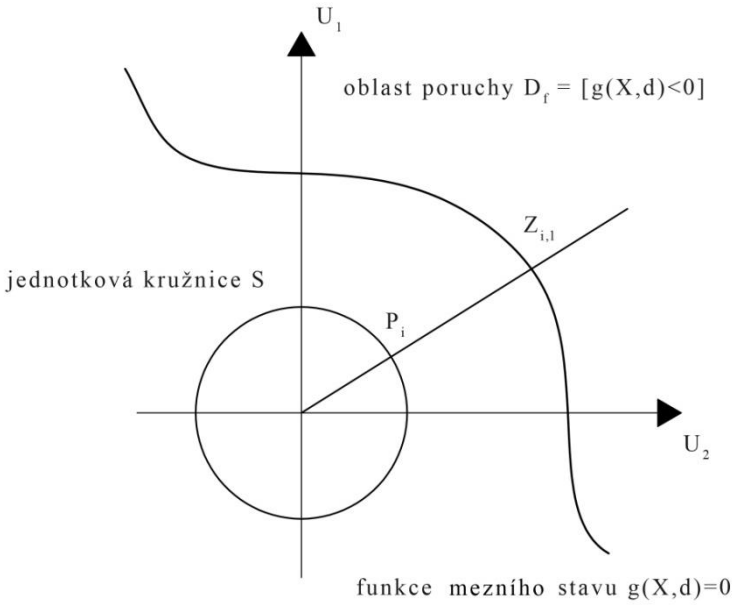
DIRECTIONAL SAMPLING

Directional sampling je další ze zdokonalených simulačních metod. Podobně jako metody FORM a SORM využívá tzv. iso-pravděpodobnostní transformace návrhového prostoru. Na rozdíl od těchto metod si však zachovává charakter metody simulační a nelze ji považovat za metodu aproximační. V transformovaném prostoru je možné (transformované) náhodné proměnné považovat za nezávislé standardní proměnné se střední hodnotou rovnou 0 a jednotkovou směrodatnou odchylkou.

Zjednodušeně lze postup simulace metodou Directional sampling přiblížit takto:

V transformovaném návrhovém prostoru definujme kružnici (kouli pro obecně n dimenzí) S se středem v počátku souřadnic a jednotkovým poloměrem. Určeme bod P_i, jež je náhodně generovaným bodem na obvodu kružnice (povrchu koule) dle rovnoměrného rozdělení. Ve směru polopřímky vedené z počátku souřadnic transformovaného náhodného prostoru bodem P_i, hledáme řešení rovnice $g(X,d) = 0$, kde $g(X,d)$ je funkcí mezního stavu závislou na vektoru náhodných proměnných X a vektoru deterministických hodnot d. Hledáme tedy bod na hranici oblasti poruchy D_f. Množina hodnot nacházejících se na dané polopřímce patřící do oblasti D_f je značena jako I_i a je podmnožinou návrhového prostoru R. Dále je vypočtena pravděpodobnost $q_i = P(\|U\| \in I_i)$, kde $\|U\|$ je transformovaná standardní náhodná veličina. Nakonec je vypočtena pravděpodobnost poruchy po N simulacích dle vztahu (24). Postup je ilustrován na Obr. 5 pro dvourozměrný případ.

$$P_{f,DS} = \frac{1}{N}\sum_{i=1}^{N} q_i \tag{24}$$



Obr. 5 Schéma průběhu metody Directional sampling ve 2D

Hodnota $Z_{i,l}$ na Obr. 5 je řešením rovnice $g(X,d) = 0$, potom platí, že body nacházející se na vyobrazené polopřímce za tímto bodem patří do množiny I_i. [49]

Výše uvedený popis je pouze zjednodušeným přiblížením podstaty metod typu Directional Sampling. Bližší informace k těmto postupům lze nalézt např. v [50].

1.4 OPTIMALIZAČNÍ TECHNIKY

Následující část textu se věnuje stochastickým optimalizačním metodám, jež umožňují s určitou pravděpodobností nalézt globální minimum obecné funkce. V současné době existují dvě hlavní skupiny nedeterministických optimalizačních metod [51]:

Metody heuristické

- Horolezecké algoritmy
- Simplexový algoritmus (Nelder – Mead)

Metody meta-heuristické (většinou stochastické metody)

- Simulované žíhání
- Genetické algoritmy
- Evoluční strategie
- Evoluční programování

Zaměřme se nyní na druhou zmiňovanou skupinu metod, tedy metody meta-heuristické.

1.4.1 METODA SIMULOVANÉHO ŽÍHÁNÍ

Metoda Simulovaného žíhání je přibližný stochastický algoritmus vycházející z Boltzmannova pravděpodobnostního rozdělení (vztah (25)) [29] a [3].

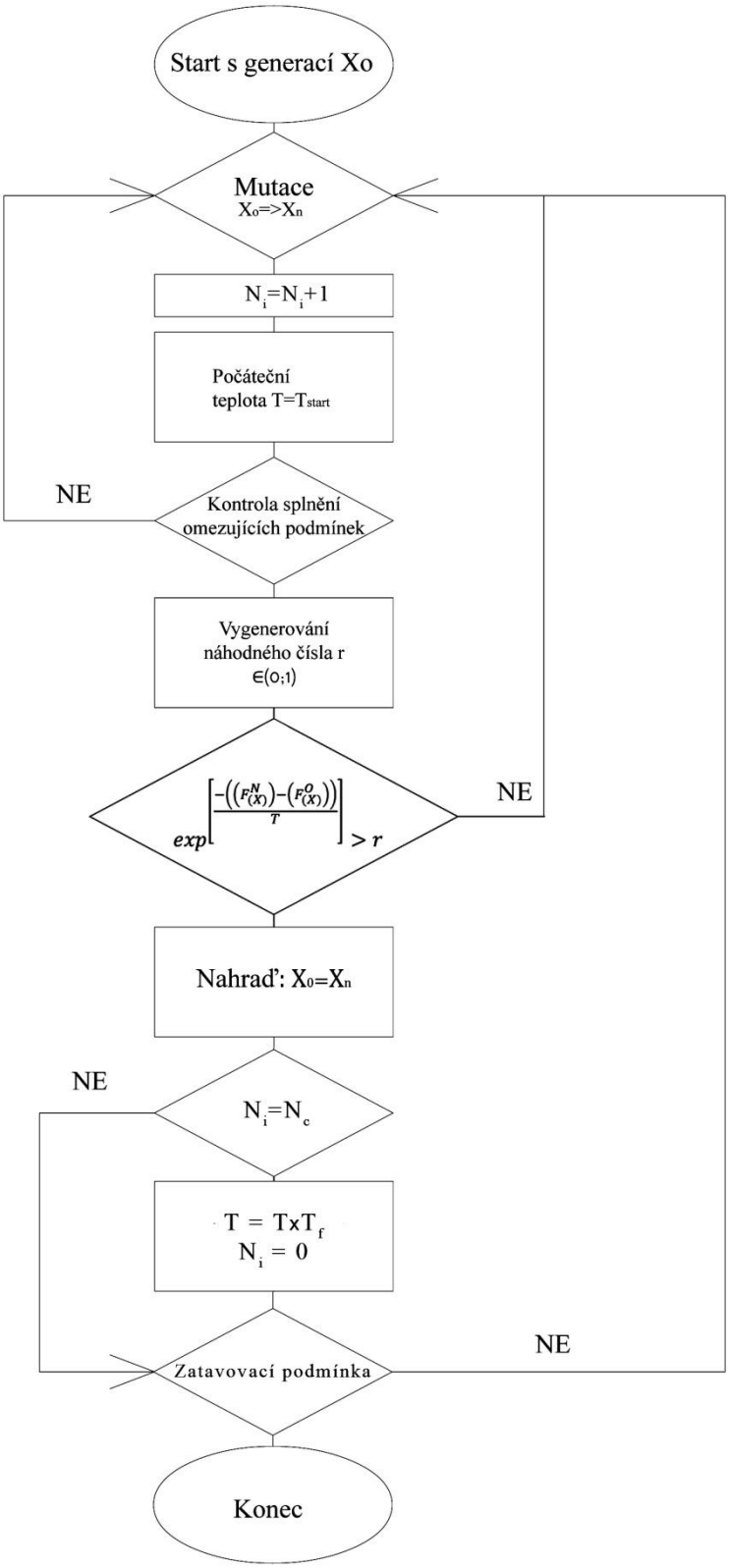
$$P_r(E) \sim e^{(-E/k_B T)} \tag{25}$$

kde E je energetická hladina, T je teplota systému a k_B je Boltzmannova konstanta, jež zavádí souvislost mezi teplotou a energií.

Tento algoritmus byl převzat z metalurgie a vychází z předpokladu, že systém v teplotní rovnováze o dané teplotě T má veškerou svou tepelnou energii rozloženou pravděpodobnostně mezi všechny různé energetické hladiny E. S určitou pravděpodobností (velmi malou) může být systém i při nízkých teplotách lokálně ve vyšším energetickém stavu. Tuto vlastnost můžeme využít při rozhodování, zda nově vygenerovaný (některým z algoritmů mutace) náhodný vektor X_n přijmeme v následující generaci jako vektor rodičovský X_o či nikoli. Aplikace metody Simulovaného žíhání umožňuje vyhledávacímu procesu „vyskočit“ z lokálního minima funkce a pokračovat směrem k minimu globálnímu.

Podstatu Simulovaného žíhání lze demonstrovat na základě energetického grafu (Obr. 7). Na obrázku je znázorněno minimum lokální (A) a globální (B). Najde-li vyhledávací algoritmus lokální minimum, pak v případě deterministických metod v něm vyhledávání většinou končí. U metody Simulovaného žíhaní existuje určitá pravděpodobnost (25), že vyhledávaní “vyskočí“ z minima lokálního a nasměruje se k minimu globálnímu. Systém však

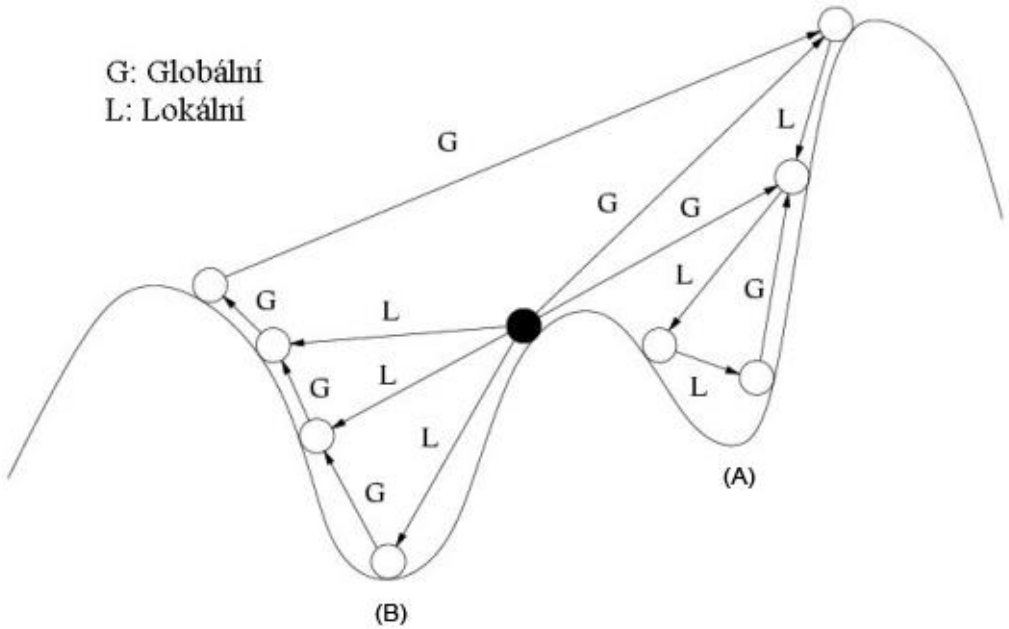
musí být dostatečně excitován (mít dost vysokou teplotu), aby energie potřebná k "vyskočení" byla dostatečná. Algoritmus metody Simulovaného žíhání je na Obr. 6. [9]



Obr. 6 Algoritmus metody Simulovaného žíhání

$F_{(X)}^{N}$ je funkční hodnota v nově vygenerovaném bodu návrhového prostoru, $F_{(X)}^{O}$ je funkční hodnota rodičovského boudu návrhového prostoru. Počáteční teplota T_{start} se podle [30] nastavuje heuristicky tak, aby po provedení $n_x 10$ iterací byl poměr úspěšných ke všem iteracím cca 0,5. Snížení teploty T teplotním redukčním faktorem T_f se provádí pravidelně po stanoveném počtu iterací N_c. N_i je čítač iterací s konstantní teplotou.

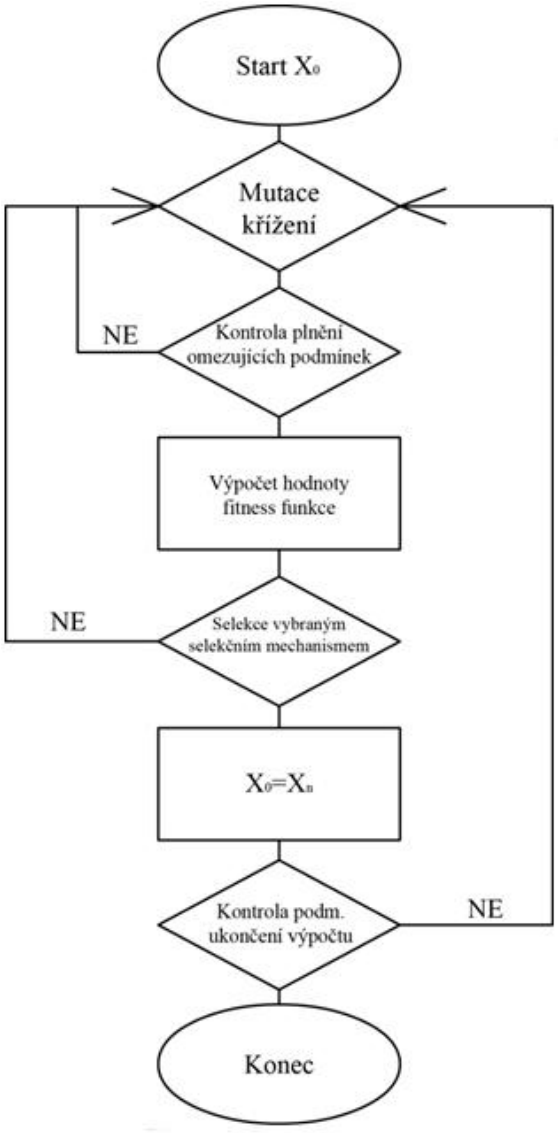
Kritérium ukončení iteračního procesu může být definováno různými způsoby, např. jako dosažení maximálního počtu iterací N_{max}, tzv. "ochlazení" na stanovenou minimální teplotu T_{min} nebo počet po sobě jdoucích zamítnutí vektoru X_n. Existuje také mnoho generátorů vektoru X_n na základě rodičovského vektoru X_o. Rovněž je k dispozici několik možných algoritmů ochlazování [51]. Volba ideálního nastavení algoritmu simulovaného žíhání tak, aby konvergoval dostatečně rychle a byl zároveň schopen najít globální extrém před dosažením podmínky ukončení iteračního procesu, je náročným úkolem řešitelným jen heuristicky. Existují sice důkazy, že při vhodném ochlazovacím schématu metoda simulovaného žíhání konverguje k nalezení globálního minima funkce, ale vzhledem k vysokému počtu simulací, se kterým metoda simulovaného žíhání pracuje, a nutnosti provést další sadu simulací k určení optimálního nastavení není tato metoda vhodná pro SSA.



Obr. 7 Energetický graf [9]

1.4.2 METODY EVOLUČNÍCH STRATEGIÍ

Metody evolučních strategií [31], [32] využívají algoritmů, jež imitují přírodní výběr k meta-heuristickému řešení výpočetně náročných problémů. Proces pracuje s náhodnými vektory X_o v tzv. generaci rodičů (první generaci X_{o1} je nutné zadat, často simulací typu Monte Carlo) a vektory X_n tzv. generace potomků. Tato je získána mutací či vzájemnou kombinací nebo křížením z generace rodičů. Pro náhodné vektory X_n je následně vypočtena hodnota jejich fitness funkce (určuje šanci vektoru X_n na přijetí do další generace). Poté je provedena selekce řešení podle zvoleného selekčního mechanismu (blíže ve [34]). Náhodné vektory X_n, jež mají nejlepší předpoklady přežít (nejlepší hodnoty fitness funkce vybrané zvoleným selekčním mechanismem), jsou v následujícím kroku přijaty do generace rodičů. Celý proces se iterativně opakuje a řešení se postupně zlepšují, dokud není splněna podmínka ukončení procesu iterace. [9]



Obr.8 Obecný algoritmus metody evolučních strategií [9]

Metody evolučních strategií jsou dnes nejvíce rozvíjenými meta-heuristickými technikami [35]. Tyto metody existují v sériové i paralelní verzi. Podle způsobu vytváření nové generace bodů návrhového prostoru rozlišujeme dva základní typy těchto metod. V *(μ, λ)-ES* je nová generace odvozena z potomků předchozích generací a v typu *(μ+λ)-ES* je nová generace výsledkem sjednocení rodičů a potomků generací předchozích. Podobně jako u metody simulovaného žíhání existuje i zde řada různých možností formulace kritéria ukončení iteračního procesu (např. limitní počet simulací či generací, dosažení určité postačující hodnoty apod.).

Počet simulací, se kterými se v rámci genetických algoritmů pracuje, je obecně nižší než u metody simulovaného žíhání, zároveň zde částečně odpadá problém heuristického hledání optimálního nastavení metody. Další výhodou těchto metod je jejich snadná paralelizace (možná několika způsoby [34]), jež umožňuje další zefektivnění vyhledávácího algoritmu z pohledu přesnosti řešení a výpočetních nároků (blíže např. v [34], [38]). Genetické algoritmy tedy představují skupinu efektivních metod využitelných pro SSA.

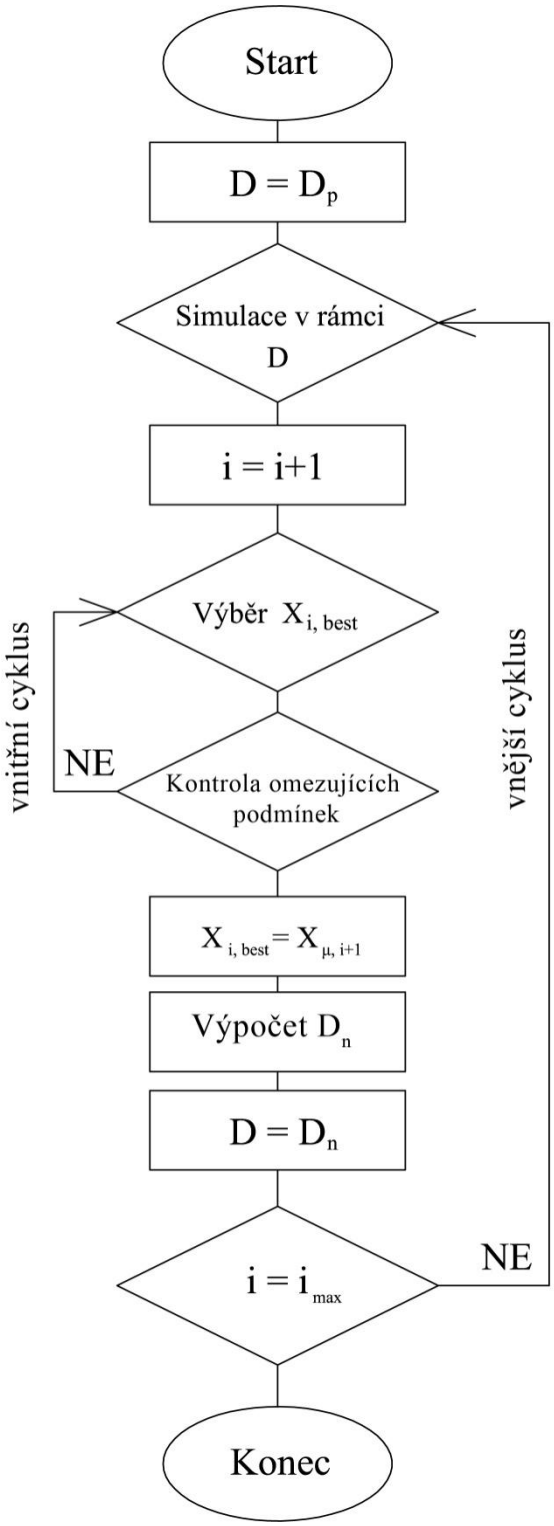
1.4.3 CÍLENÉ VÍCEÚROVŇOVÉ VZORKOVÁNÍ (AIMED MULTILEVEL SAMPLING)

Nejjednodušší metodou heuristické optimalizace je provedení simulace typu Monte Carlo v rámci návrhového prostoru a následný výběr nejvíce vyhovující realizace (vzhledem k daným kritériím optimalizace. Takový postup jednoznačně nekonverguje směrem k optimu funkce a kvalita řešení se odvíjí od počtu provedených simulací. Přesná lokalizace optima je při využití pouze prostých simulačních metod vysoce nepravděpodobná. Rozptyl výsledků takové optimalizace je v případě SSA velmi vysoký a silně závislý na počtu provedených simulací. Tento postup je však velice jednoduchý, nevyžaduje žádné znalosti topologie optimalizované funkce a je z inženýrského pohledu transparentní a snadno aplikovatelný.

Metoda cíleného víceúrovňového vzorkování byla poprvé nastíněna v [9] (pod názvem Nested LHS). Její základní myšlenkou je seřadit průběh simulace do několika úrovní. Na každé úrovni poté proběhne vzorkování v rámci definovaného prostoru. Následně bude vybrán vzorek s nejlepšími vlastnostmi vzhledem k definici optimalizačního problému. Návrhový vektor $X_{i,\ best}\ (x_1;x_2;...;x_n)$ odpovídající nejlepšímu, v *i*-té úrovni vygenerovanému vzorku, je určen jako vektor středních hodnot náhodných veličin pro simulaci v rámci další úrovně. Následně je "zmenšen" vzorkovací prostor okolo nejlepšího vzorku. V tomto zmenšeném prostoru pak probíhá další simulace. Dochází tak ke stále podrobnějšímu prohledávání oblastí okolo vzorků s nejlepšími vlastnostmi. Obecný algoritmus představené metody je popsán ve vývojovém diagramu na Obr. 9.

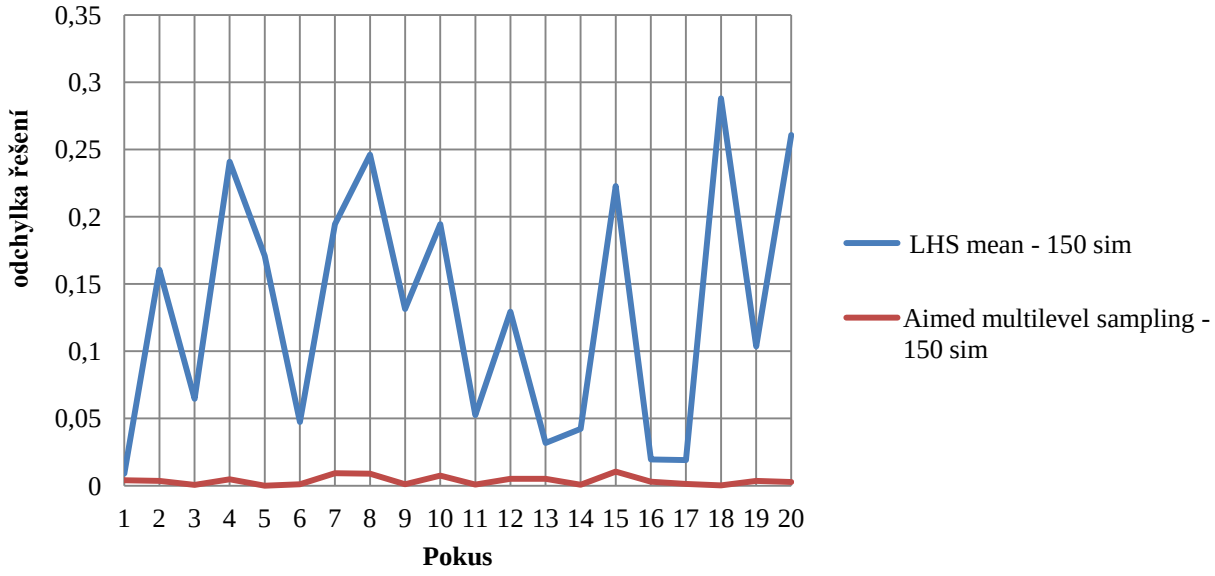
Tato metoda si zachovává jednoduchost prostého vzorkování, zároveň je však schopna konvergence směrem k optimu cílové funkce. Rozptyl výsledků optimalizace touto metodou je pak mnohonásobně menší než v případě prostého vzorkování. Byla provedena řada testů k určení rozptylu řešení pro neohraničenou optimalizaci obecné funkce [52] (viz kapitola 3.1). Příklady srovnání výsledků cíleného víceúrovňového vzorkování a prosté simulace se stejným

celkovým množstvím simulací při optimalizaci vybraných testovacích funkcí jsou zobrazeny v grafech 1 a 2.

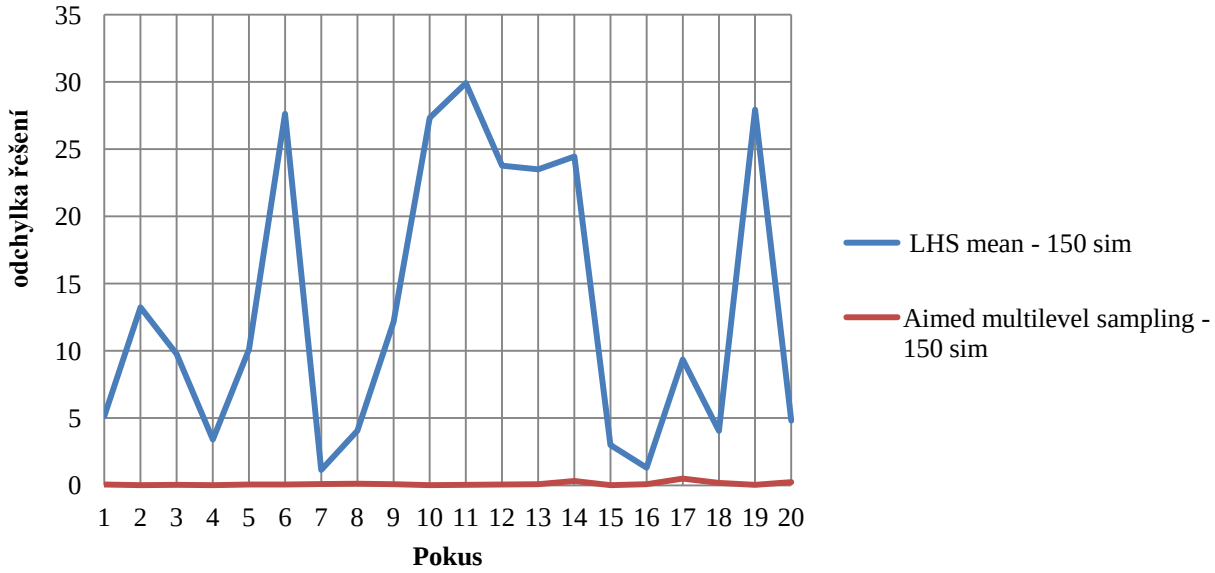


Obr. 9 Obecný algoritmus Aimed Multilevel Sampling

Hodnota D na Obr. 9 představuje vzorkovací prostor o n dimenzích. D_p je pak počáteční návrhový prostor optimalizační úlohy. D_n je "zmenšený" vzorkovací prostor pro simulaci na úrovni $i+1$. Čítač úrovní je reprezentován hodnotou i, kde i_{max} je maximální počet úrovní sloužící jako kritérium ukončení procesu. Vektor $X_{\mu,\ i+1}$ je vektorem středních hodnot náhodných veličin n-rozměrného návrhového prostoru D pro simulaci v úrovni $i+1$.



Graf 1 Srovnání odchylky od optima u prosté simulace metodou LHS a Cíleného víceúrovňového vzorkování pro Six-hump camel back function.

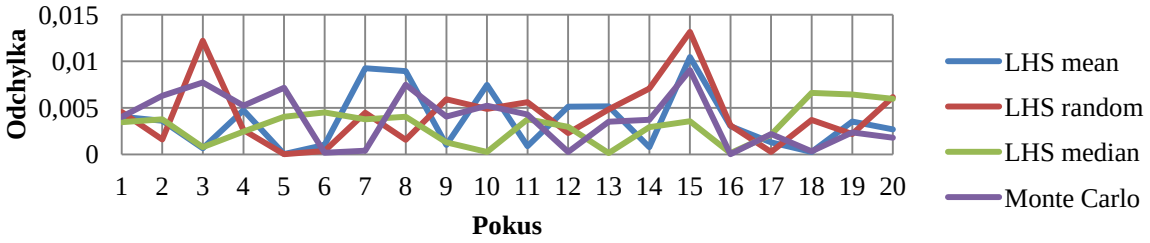


Graf 2 Srovnání odchylky od optima u prosté simulace metodou LHS a Cíleného víceúrovňového vzorkování pro Goldstein-Price's function.

Pro sestavení grafů 1 a 2 bylo provedeno dvacet náhodných optimalizačních pokusů (vždy s jiným nastavením zdrojového čísla pro generátor náhodných čísel) s užitím jak prosté simulace metodou LHS mean, tak Cíleného víceúrovňového vzorkování (dále jen AMS – Aimed Multilevel Sampling). V případě obou metod bylo generováno celkem 150 simulací. U AMS simulace probíhala v 5 úrovních s 30 simulacemi na každé z nich. Vzorkovací prostor byl v každé další úrovni redukován na polovinu své předešlé hodnoty. Testy byly provedeny s pomocí akademického softwaru FNPO popsaného v kapitole 2.2. Bližší informace o užitých testovacích funkcích jsou k dispozici v kapitole 3.1 a [52].

Z grafů 1 a 2 je zřejmé, že je algoritmus AMS schopen lépe konvergovat ke globálnímu optimu funkce. Střední hodnota odchylky od optima byla u Six-hump camel back function *39,8x* a u Goldstein-Price's function *109,3x* nižší při užití AMS než u prosté simulace LHS mean. Z grafů je také patrná nízká hodnota rozptylů výsledků při použití metody AMS. Rozptyl výsledků i hodnoty nalezeného optima jde dále vylepšit specifickým nastavením algoritmu AMS (viz dále). Podobného výsledku bylo dosaženo i u dalších testovacích funkcí.

Pro simulaci v rámci algoritmu AMS jsou vhodné metody LHS či klasická metoda Monte Carlo. Pro potřeby optimalizace se většinou volí rovnoměrné rozdělení pravděpodobnosti, které nediskriminuje žádné z hodnot v rámci definičního oboru náhodné proměnné. Není však vyloučeno, že v některých případech může mít preference realizací okolo středních hodnot, daná volbou jiného než rovnoměrného rozdělení, pozitivní vliv na průběh optimalizace. Určení vlivu užitého rozdělení pravděpodobnosti na výsledek optimalizace bude cílem některého z dalších testů. V případě standardní volby rektangulárního rozdělení pravděpodobnosti náhodných proměnných je zbytečné provádět simulace výpočetně náročnější metodou LHS mean (nutnost řešit integrál (17)), která pro toto rozdělení dává stejné simulace jako méně náročná metoda LHS median. Během testování účinnosti jednotlivých simulačních metod užívaných v rámci AMS se ukázalo, že samotná volba metody (při užití rovnoměrného rozdělení náhodných veličin) nemá na výsledek optimalizace velký vliv. Jako mírně účinnější v porovnání s ostatními metodami se jeví metoda LHS median. Vzhledem k nízkému počtu doposud provedených testů může být tento závěr ovlivněn statistickou chybou. Srovnání odchylek od optima při dvaceti testech provedených na Six-hump camel back function metodou AMS s užitím různých simulačních metod je zobrazeno v grafu 3.



Graf 3 Srovnání simulačních metod užitých v AMS (Six-hump camel back function)

Algoritmus AMS umožňuje uživateli kontrolovat průběh celého procesu. Je možné jej přerušit na každé z úrovní a upravit jeho nastavení na základě výsledků dosažených na úrovních předchozích. Toto je obzvláště efektivní v případech, kdy je vyčíslení funkční hodnoty realizace náhodného vektoru časově mnohem náročnější než samotné hodnocení a přenastavení algoritmu. AMS nabízí následující výčet nastavitelných parametrů:

- Výběr simulační metody (LHS mean, median, random, Monte Carlo)
- Počet úrovní (cyklů) algoritmu
- Počet simulací na každé z úrovní
- Změna velikosti vzorkovacího prostoru na každé další úrovni pro každou z proměnných definovatelná zvlášť (např. na základě analýzy sensitivity)
- Funkce hustoty rozdělení pravděpodobnosti definovatelná pro každou z proměnných

Na začátku procesu, kdy uživatel většinou nemá povědomí o průběhu optimalizované funkce, je vhodné volit počet simulací v rámci úrovně vyšší a tempo zmenšování vzorkovacího prostoru pomalejší. Naopak v momentě, kdy je algoritmus "nasměrován" k určitému řešení, je možné počet simulací na jednotlivých úrovních snížit a tempo zaměřování urychlit, čímž dosáhneme rychlejší konvergence. Zvýší se však riziko, že vyhledávání uvízne v minimu lokálním nebo dojde k vyčerpání konvergenčního potenciálu algoritmu dříve, než bude extrému dosaženo (viz dále). Vstupování do procesu na jeho jednotlivých úrovních může přinést určité výhody, je však v rozporu s myšlenkou plné automatizace procesu optimalizace. Není vyloučen budoucí vývoj softwarových prostředků schopných hodnotit dosavadní průběh procesu a reagovat na něj změnou nastavení výše zmíněných parametrů algoritmu AMS. Prozatím však software FNPO popisovaný v kapitole 2.2 ponechává možnost vstupu do algoritmu na některé z jeho úrovní pouze na uživateli. Následující odstavce se věnují způsobu obecného nastavení parametrů algoritmu AMS tak, aby byl umožněn jeho plně automatický průběh s uspokojivou účinností pro širokou skupinu optimalizovaných funkcí. Míra uživatelského zásahu pak závisí čistě na rozhodnutí uživatele na základě jeho znalosti konkrétní optimalizované funkce.

KONVERGENČNÍ RYCHLOST A OPTIMÁLNÍ NASTAVENÍ AMS

Vyjděme z předpokladu, že maximální počet simulací je konstantně volená hodnota daná vztahem:

$$N_{sim} = N_c i_{max} \tag{26}$$

kde N_{sim} je zvolený maximální počet simulací daný součtem simulací na všech úrovních, i_{max} je počet úrovní (cyklů) a N_c je počet vzorků na každé z úrovní. Nemáme-li představu o tvaru a vlastnostech cílové funkce, je vhodné volit způsob zmenšování vzorkovacího prostoru tak, že nový vzorkovací prostor vznikne násobením rozměrů návrhového prostoru předešlé úrovně číslem q z intervalu *(0; 1)*. Velikost vzorkovacího prostoru na každé z úrovní tedy odpovídá

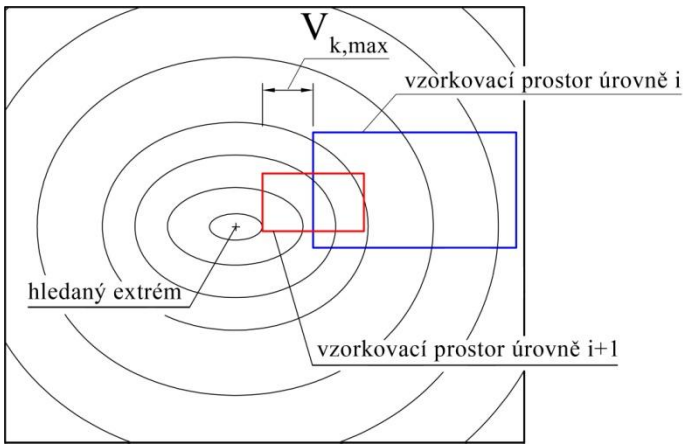
jednotlivým členům konvergentní geometrické řady. Velikost každé z dimenzí vzorkovacího prostoru i-té úrovně a_i pak bude dána vztahem (27).

$$a_i = a_1 q^{i-1} \tag{27}$$

kde a_1 je počáteční velikost návrhového prostoru daná definicí návrhového prostoru konkrétní cílové funkce.

Představme si nyní situaci kdy hodnota i_{max} je vysoká a současně hodnota q je blízká 0. Pro i blížící se i_{max} tedy dochází ke stavu, kdy je vzorkovací prostor již velice malý. Jestliže se v každé další úrovni může vyhledávání posunout jen o maximálně polovinu rozměru vzorkovacího prostoru úrovně následující (směrem k optimu pro každý z rozměrů návrhového prostoru), pak se pro i blížící se i_{max} a nevhodně zvolenou hodnotu q bude rychlost konvergence asymptoticky blížit 0 - dojde k vyčerpání konvergenčního potenciálu algoritmu. Budou tedy generovány další vzorky, aniž by docházelo ke zlepšení doposud nalezené hodnoty. Abychom zamezili plýtvání výpočetním výkonem, musíme správně zvolit kombinaci hodnot N_c, i_{max} a q, ta je pro užití algoritmu AMS klíčová.

Zaveďme nyní tzv. maximální teoretickou rychlost konvergence $v_{k,max}$, jež je definována jako vzdálenost, o níž se může vyhledávání maximálně posunout v každém z n rozměrů návrhového prostoru ve směru optima cílové funkce na následující úrovni algoritmu AMS. Grafická interpretace této veličiny je na Obr. 10.



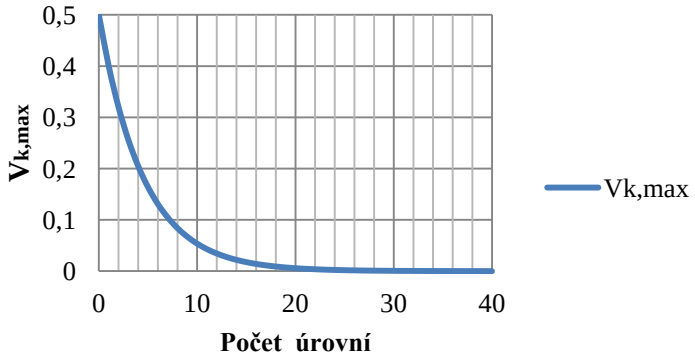
Obr. 10 Grafická interpretace maximální teoretické rychlosti konvergence $v_{k,max}$ pro q = 0,5

Vztah pro výpočet maximální teoretické rychlosti konvergence $v_{k,max}$, je možné odvodit z Obr. 10 a vztahu (27). Maximální posun vzorkovacího prostoru směrem k hodnotě funkčního optima vzhledem k jednotlivým rozměrům návrhového prostoru nastane teoreticky tehdy, pokud se na úrovni i vygeneruje nejvíce vyhovující realizace na hranici vzorkovacího prostoru. Tento předpoklad platí pro situace, kdy je hledané optimum situováno mimo vzorkovací prostor úrovně i. V úrovni $i+1$ je pak vzorkovací prostor umístěn dle situace

zachycené na Obr 10. Hodnota $v_{k,max}$ tedy odpovídá polovině rozměru vzorkovacího prostoru úrovně $i+1$. Ze vztahu (27) plyne, že velikost vzorkovacího prostoru úrovně $i+1$ lze vypočíst jako: $a_{i+1} = a_1 q^i$. Hodnota veličiny $v_{k,max}$ na úrovni i je pak dle výše uvedené definice dána vztahem (28).

$$v_{k,max} = \frac{a_1 q^i}{2} \tag{28}$$

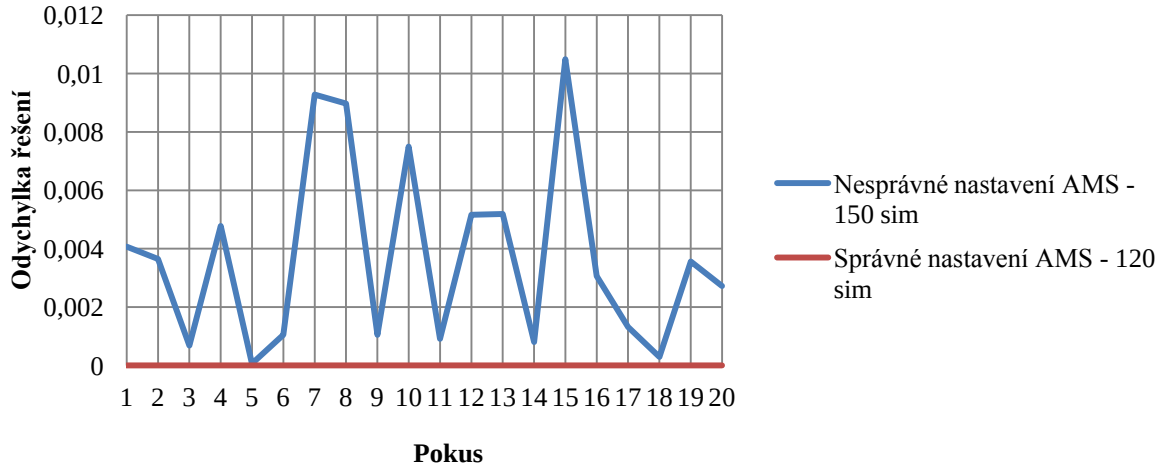
Grafy 4 a 5 demonstrují průběh $v_{k,max}$ v závislosti na počtu úrovní pro různé hodnoty q. Je zřejmé, že v závislosti na volbě hodnoty q musíme zvolit odpovídající rozumnou hodnotu i_{max}. Např. z grafu 4 vyplývá, že pro $q = 0{,}8$ je vhodné volit počet úrovní $i_{max} = 22$. Naopak graf 5 ukazuje, že pro $q = 0{,}5$ je vhodné volit $i_{max} = 8$. Grafy 4 a 5 zobrazují normovanou hodnotu $v_{k,max}$, tedy hodnotu pro $a_1 = 1$, kde a_1 je základní velikost návrhového prostoru. Je evidentní, že hodnota $v_{k,max}$ se při zmenšování návrhového prostoru snižuje prakticky až k nule – pak nemá cenu pokračovat v průběhu algoritmu AMS. Maximální počet úrovní metody AMS se vzhledem k volbě parametru q a původní velikosti jednotlivých rozměrů návrhového prostoru a_1 (definované v zadání daného optimalizačního problému) stanoví na základě grafu průběhu $v_{k,max}$ v závislosti na počtu úrovní.



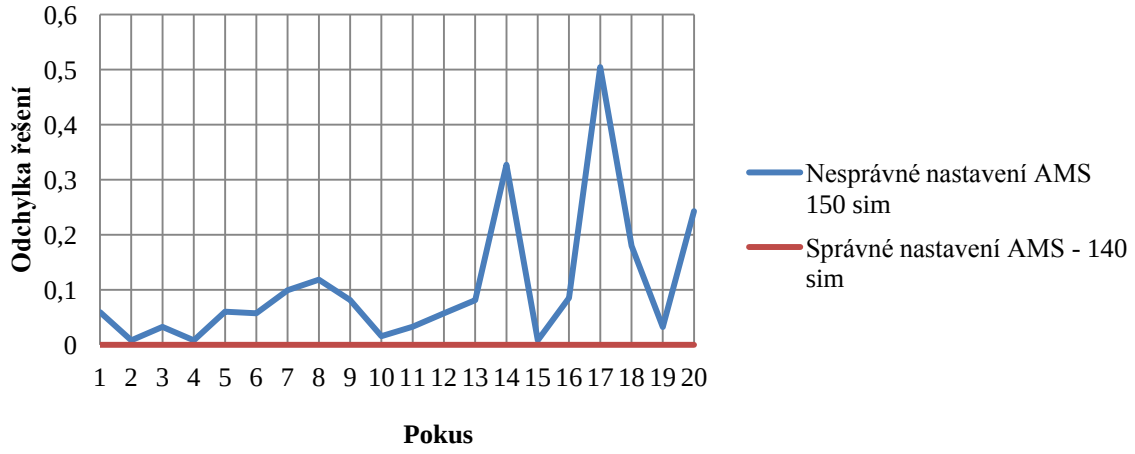
Graf 4 Závislost $v_{k,max}$ na počtu úrovní pro q = 0,8

Graf 5 Závislost $v_{k,max}$ na počtu úrovní pro q = 0,5

Význam zavedení veličiny $v_{k,max}$ a z ní plynoucí optimální volby počtu úrovní pro danou hodnotu q je nejlépe patrný při srovnání výsledků optimalizace za správného a nesprávného nastavení i_{max} vzhledem k použité hodnotě q. V grafech 1 a 2 bylo uvedeno srovnání optimalizace pomocí prosté simulace metodou LHS a algoritmu AMS. S ohledem na výše popsané správné nastavení algoritmu AMS je evidentní, že při testu zachyceném v grafech 1 a 2 je volba parametru $i_{max} = 5$ pro $q = 0{,}5$ a $N_{sim} = 150$ chybná (není dosaženo plného konvergenčního potenciálu metody AMS). Přesto dosahuje AMS lepších výsledků než prostá simulace metodou LHS. Uvážíme-li závislost optimálního počtu úrovní algoritmu AMS na volbě parametru $q = 0{,}5$ danou vztahem (28), pak ze zmíněného vztahu plyne, že např. pro Six-hump camel back function při $N_{sim} = 120$ bude ideální $i_{max} = 12$ a $N_c = 10$. Porovnání výsledků za správného a nesprávného nastavení AMS při 20 optimalizačních pokusech je zachyceno v grafech 6 a 7.

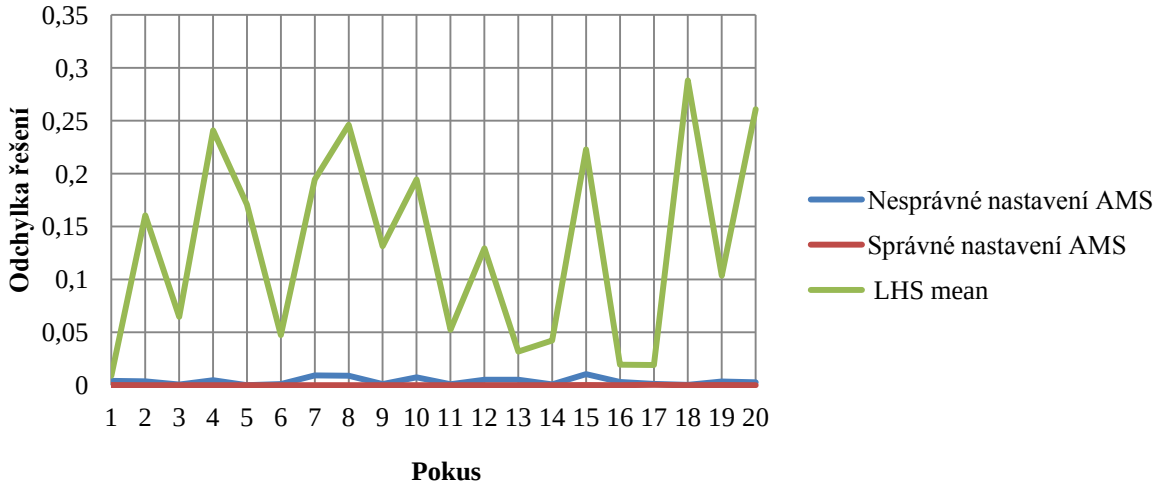


Graf 6 Srovnání správného a nesprávného nastavení AMS při optimalizaci Six-hump camel back function



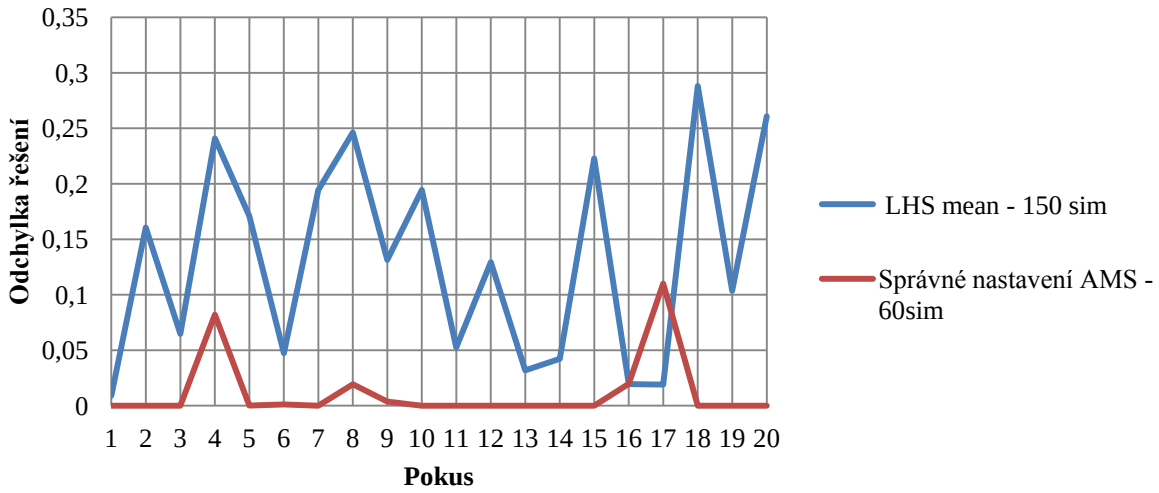
Graf 7 Srovnání správného a nesprávného nastavení AMS při optimalizaci Goldstein-Price's function

Grafy 6 a 7 tedy dokládají zásadní význam vztahu (28) pro účinné nastavení algoritmu AMS. Při správném nastavení algoritmu AMS bylo minimum úspěšně nalezeno s přesností na 6 desetinných míst u obou zmíněných funkcí při všech 20 testech. Pro Six-hump camel back function bylo ke zmíněné přesnosti potřeba 120 simulací, u Goldstein-Price's function pak bylo zapotřebí 140 simulací.



Graf 8 Porovnání správného, nesprávného nastavení AMS a prosté simulace LHS (Six-hump camel back function)

Znalost správného nastavení AMS otevírá prostor k možnému snížení počtu simulací potřebných k lokalizaci extrému. Při nižším počtu simulací se algoritmu také daří přesněji lokalizovat extrém, se snižujícím se počtem simulací však klesá pravděpodobnost, že proces optimalizace bude úspěšný. Graf 9 ukazuje porovnání optimalizace Six-hump camel back function prostou simulací LHS s užitím 150 simulací a algoritmem AMS, který pracuje s pouhými 60 simulacemi, za optimálního nastavení dle (28).



Graf 9 Srovnání LHS – 150 simulací a AMS – 60 simulací (Six-hump camel back function)

Z grafu 9 je patrné, že při užití pouhých 60 simulací se při aplikaci AMS na Six-hump camel back function podařilo nalézt extrém s přesností na 2 desetinná místa v 16 z 20 optimalizačních pokusů. Střední hodnota odchylky pak byla pouze 0,00003. Směrodatná odchylka činila 0,02901077.

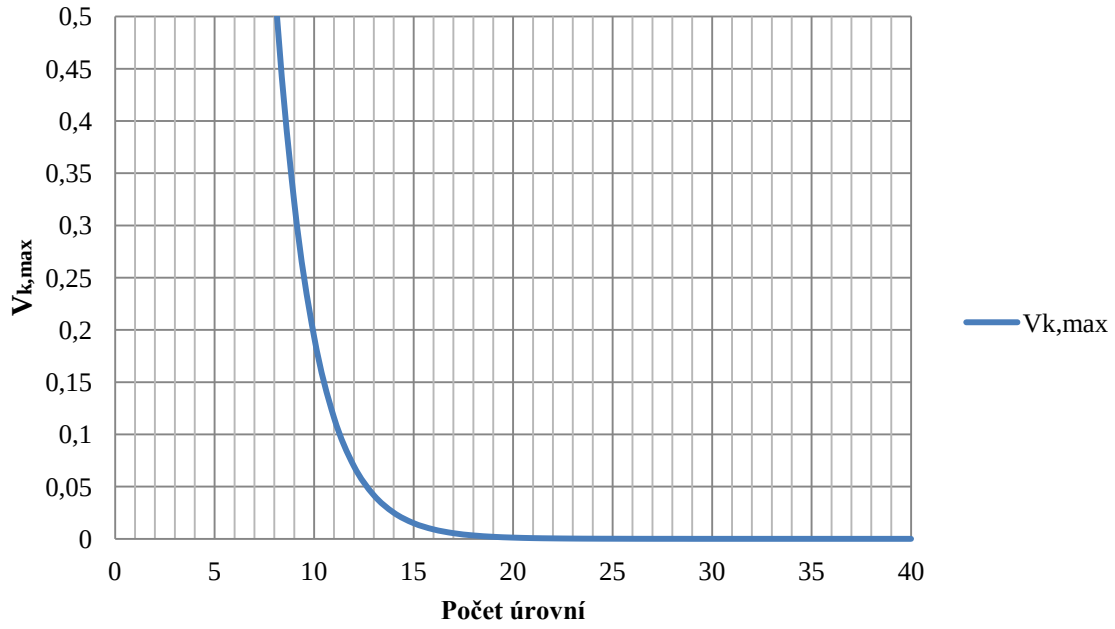
Výše uvedené grafy potvrzují schopnost algoritmu AMS konvergovat směrem ke globálnímu extrému funkce. Další testy této metody spolu se zdrojovými tabulkami uvedených grafů jsou uvedeny v kapitole 3.2.

Během testování algoritmu AMS na funkcích vyšších dimenzí se projevila nedokonalost vztahu (28). Zmíněný vztah platí přesně pouze v jednorozměrném prostoru, v němž je návrhový vektor dán pouze jedním vstupním parametrem. S rostoucím počtem rozměrů návrhového prostoru roste zároveň počet možných variací vstupních parametrů návrhového vektoru. K nalezení optima funkce je třeba identifikovat tyto vstupní parametry s co největší přesností. Při velkém počtu rozměrů je pak i v malém návrhovém prostoru na poslední úrovni AMS velké množství možných variací vstupních parametrů, jež mohou odpovídat širokému rozptylu funkčních hodnot. Jinými slovy nedojde k dostatečně "přesnému" zacílení algoritmu. Tento jev je možné demonstrovat na příkladu optimalizace Ackley's function [52] o deseti dimenzích. Srovnáme-li účinnost algoritmu AMS nastaveného dle vztahů (28) a (29), zjistíme, že při použití vztahu (29) došlo k redukci střední hodnoty chyby řešení přibližně o dvě třetiny. Je tedy zřejmé, že ve vztahu pro konvergenční rychlost musí být zohledněn také počet dimenzí řešené úlohy. Není také vyloučena souvislost mezi počtem dimenzí a optimálním počtem simulací v každém z cyklů. Přesná role počtu dimenzí při optimálním nastavení algoritmu AMS zatím nebyla určena. Vliv počtu dimenzí na nastavení AMS byl zatím stanoven pouze heuristicky zavedením součinitele n, který odpovídá počtu dimenzí optimalizační úlohy do vztahu (28).

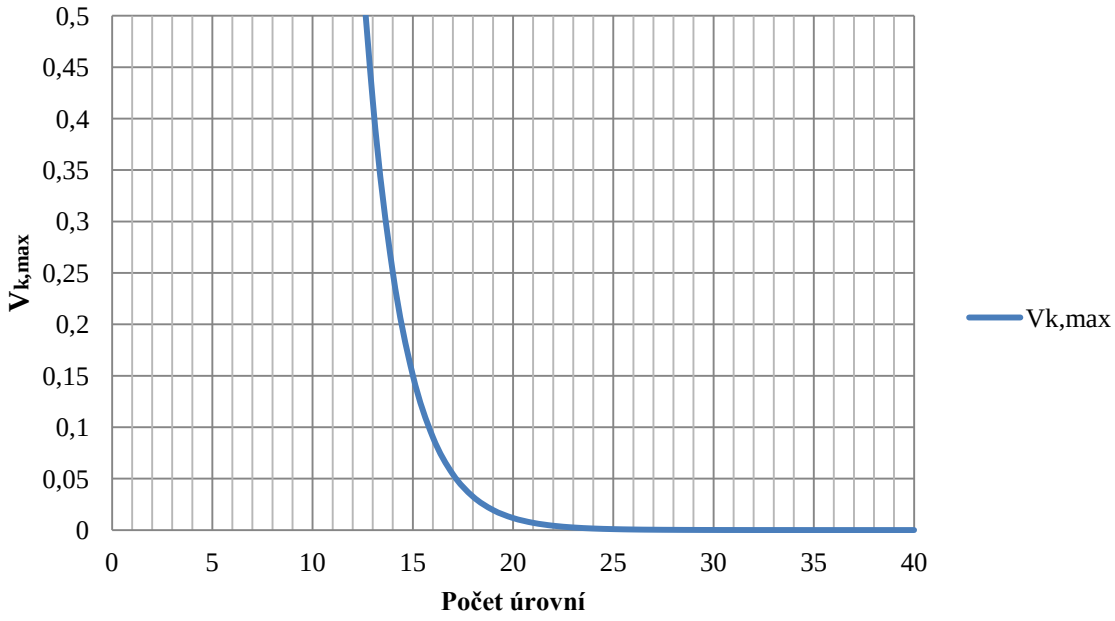
$$v_{k,max} = n\frac{a_1 q^i}{2} \tag{29}$$

Při aplikaci vztahu (29) na nastavení AMS pro řešení výše zmíněné optimalizace Ackley's function o deseti rozměrech bylo při provedení 20 optimalizačních pokusů s celkovým počtem 27000 simulací (srovnatelný počet simulací potřebují i mnohé varianty evolučních algoritmů viz [35]) 16x lokalizováno minimum s přesností na 3 desetinná místa. Střední hodnota odchylky od správného řešení činila 0,000205239. Vztah (29) není přesně odvozeným vztahem pro správné nastavení AMS v závislosti na počtu dimenzí. Nalezení přesného vztahu pro konvergenční rychlost zohledňujícího počet dimenzí úlohy bude předmětem dalšího výzkumu. Na základě provedených testů je možné usuzovat, že přesné nastavení algoritmu AMS je s rostoucím počtem dimenzí stále důležitější. V případě dvourozměrného návrhového prostoru se optimální nastavení dle vztahu (29) téměř neliší od nastavení dle vztahu (28). V mnohodimenzionálním návrhovém prostoru je zohlednění počtu rozměrů zásadní. Grafy 10 a 11 ukazují rozdíl v průběhu konvergenční rychlosti mezi

nastavením algoritmu AMS dle vztahu (28) a (29) pro případ výše zmíněné optimalizace Ackley's function v 10D. Při nastavení dle vztahu (28) odpovídajícímu grafu 10 byla pro hodnotu $q = 0{,}6$ volena hodnota $i_{max} = 23$. U nastavení dle vztahu (29) byla pro stejnou hodnotu q volena hodnota $i_{max} = 27$. Tabulka obsahující výsledky jednotlivých testů je k dispozici v kapitole 3.2.

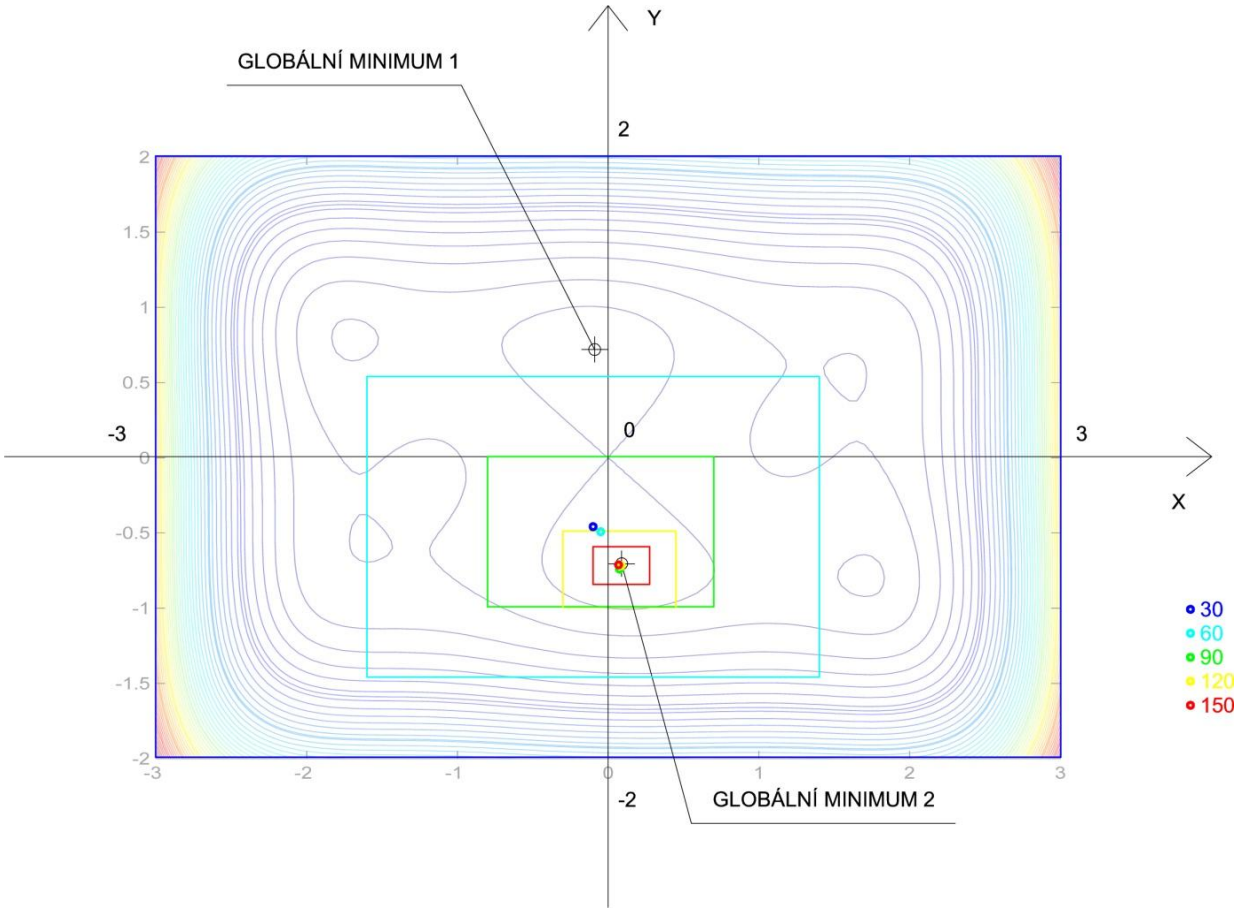


Graf 10 Závislost $v_{k,max}$ na počtu úrovní při q = 0,6 pro Ackley's function stanovená dle (28)



Graf 11 Závislost $v_{k,max}$ na počtu úrovní při q = 0,6 pro Ackley's function stanovená dle (29)

Metoda AMS tedy poskytuje stabilní řešení u prozatím otestovaných cílových funkcí. Byl navržen předpis (29) pro účinné nastavení parametrů AMS, jež je v nejjednodušším případě redukováno pouze na volbu celkového počtu simulací a parametru q. Odpadá zde nutnost složitého nastavení algoritmu jako v případě metody Simulovaného žíhání. Zároveň je však uživateli ponechána možnost podrobného nastavení AMS na základě znalostí tvaru a průběhu cílové funkce. Další výhodou popsané metody je její snadná paralelizace, jež je možná hned v několika provedeních a bude předmětem zkoumání během dalšího vývoje uvedené metody. Současně existuje možnost spojení algoritmu AMS a metody Simulovaného žíhání, kde Simulované žíhání může být využito v rámci algoritmu AMS jako výběrové kritérium nejlepší realizace dané úrovně. Algoritmus AMS by také mohl sloužit např. k vytvoření počáteční generace vzorků pro různé typy Evolučních algoritmů. Mohl by tak pro tyto vyhledávací metody poskytnout lepší výchozí generaci než standardní vzorkování např. metodami LHS. Dosavadní výsledky testování jsou slibné a naznačují velký potenciál metody AMS v rámci SSA. Počet provedených testů však zatím stále není dostatečný a identifikace účinnosti metody AMS bude dále cílem testování.



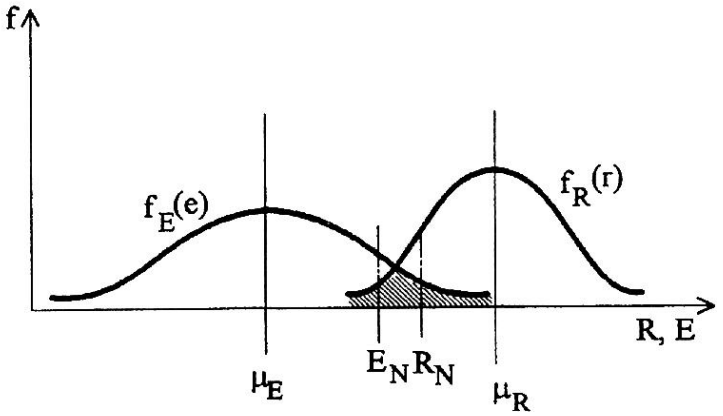
Obr. 11 průběh algoritmu AMS při optimalizaci Six-hump camel back function [9]

1.5 SPOLEHLIVOSTNÍ PODMÍNKY A PRAVDĚPODOBNOST PORUCHY

V následující části textu budou stručně nastíněny základní principy vyhodnocení pravděpodobnosti poruchy a podmínek spolehlivosti v oblasti posuzování stavebních konstrukcí, jež tvoří základní rámec pro aplikaci metod popsaných v kapitolách 1.3 a 1.6.

1.5.1 PODMÍNKA SPOLEHLIVOSTI

V procesu teoretického návrhu konstrukce existuje celá řada nejistot na všech jeho úrovních. Výsledný model je většinou vyhodnocován po částech, z nichž každá musí vyhovovat různým kriteriím spolehlivosti. Spolehlivostní podmínky však musí splnit i konstrukce jako celek. Ta musí být navržena tak, aby byla schopna odolat účinku zatížení. Definujme tedy dvě základní veličiny konstrukčního posuzování – odolnost konstrukce R a účinek zatížení E. Uvážíme-li nejistoty, jimiž je vyhodnocení obou zmíněných veličin zatíženo, bude k jejich popisu vhodné použít funkce hustoty pravděpodobnosti, tedy $f_R(r)$ *a* $f_E(e)$. Obr. 12 znázorňuje srovnání deterministické a statistické reprezentace odezvy konstrukce R a účinku zatížení E.



Obr. 12 Srovnání deterministické a statistické reprezentace veličin R a E – převzato z [1]

kde R_N a E_N představují deterministické hodnoty užívané ve standardních normových postupech (např. [53]). Z Obr. 12 je tedy patrný rozdíl mezi deterministicky formulovanou podmínkou spolehlivosti:

$$R_N \geq E_N \tag{30}$$

a pravděpodobnostním přístupem definovaným tvarem:

$$R - E \geq 0 \tag{31}$$

kde R a E jsou náhodné veličiny definované hustotami pravděpodobnosti $f_R(r)$ *a* $f_E(e)$.

Levá strana vztahu (31) je v literatuře označována jako funkce poruchy Z nebo také rezerva spolehlivosti. Selhání konstrukce v některém z mezních stavů [1] tedy nastane v případě platnosti vztahu (32).

$$R - E = Z < 0 \tag{32}$$

Rezervu spolehlivosti lze formálně definovat jako funkci náhodných veličin $\boldsymbol{X} = X1, X2, ..., Xn$ vztažených k R a E.

$$g(\boldsymbol{X}) = g(X_1, X_2, \dots., X_n) \geq 0 \tag{33}$$

Funkce poruchy reprezentující podmínku spolehlivosti tedy může být explicitně nebo implicitně definovaná funkce obecné složitosti, závislá na kombinaci n náhodných (nebo náhodných a deterministických) vstupních veličin.

1.5.2 PRAVDĚPODOBNOST PORUCHY

Základní veličinou sloužící k vyčíslení spolehlivosti je teoretická pravděpodobnost poruchy p_f definovaná dle vztahu (34). Vyšrafovaná část na Obr. 12 představuje oblast k výpočtu p_f.

$$p_f = P(R - E < 0) \equiv P(Z < 0) \tag{34}$$

Pravděpodobnost poruchy je závislá na obsahu této plochy, který je dán velikostí středních hodnot μ_R a μ_E, rozptylem veličin E, R a tvarem křivek charakterizovaným funkcemi $f_R(r)$ a $f_E(e)$.

Pravděpodobnost, že konstrukční odezva R bude menší než daná hodnota x, reprezentuje funkční hodnota distribuční funkce v bodě x $\Phi_R(x)$, pro niž platí:

$$P(R \leq x) = \Phi_R(x) \tag{35}$$

Pravděpodobnost, že se vliv zatížení E bude nacházet v infinitesimálním intervalu dx kolem bodu x je rovna ploše:

$$P\left(x - \frac{dx}{x} \leq E \leq x + \frac{dx}{x}\right) = f_E(x)dx \tag{36}$$

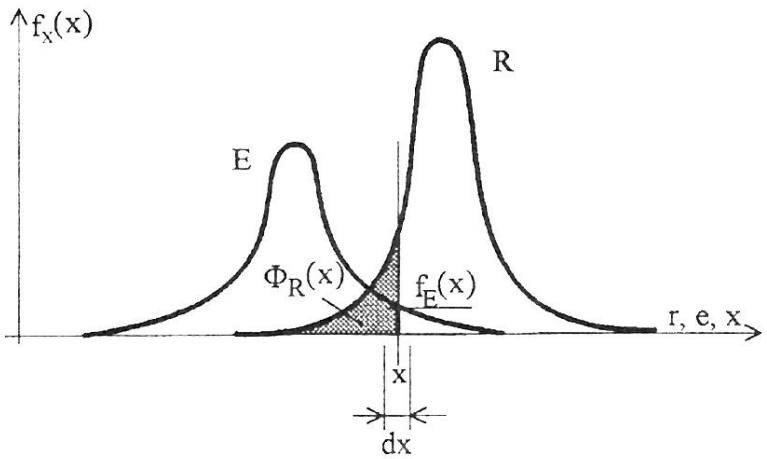
(viz Obr. 13). Pravděpodobnost, že výrazy (35) a (36) jsou splněny současně, je následně dána součinem jejich pravých stran. Uvážíme-li, že x může nabývat hodnot od $-\infty$ do ∞, dostaneme pravděpodobnost poruchy integrací dle vztahu (37).

$$p_f = \int_{-\infty}^{\infty} dp_f = \int_{-\infty}^{\infty} f_E(x)\Phi_R(x)dx \tag{37}$$

Tento na první pohled jednoduchý integrální vztah je možné řešit v uzavřené formě pouze pro určité jednoduché případy (např. s předpokladem rovnoměrného rozdělení pravděpodobnosti veličin *R* a *E*). Je-li spolehlivost definována jako doplněk pravděpodobnosti poruchy, pak pravděpodobnost poruchy samotnou můžeme kvantifikovat jako:

$$p_f = 1 - \int_{-\infty}^{\infty} f_R(x)\Phi_E(x)dx \tag{38}$$

tedy jako doplněk spolehlivosti.



Obr. 13 Výpočet pravděpodobnosti poruchy – převzato z [1]

Pro dosažení požadované úrovně spolehlivosti je potřeba, aby návrhové hodnoty odezvy konstrukce a účinku zatížení užité během návrhu klasickým přístupem byly voleny s požadavkem co nejmenší plochy překrytí křivek hustot pravděpodobnosti. Toho je při klasickém přístupu dosahováno pomocí různých koeficientů. Plně pravděpodobnostní přístup pak představuje kvalitativně vyšší úroveň návrhu. Při znalosti $f_R(r)$ a $f_E(e)$ je pak možné volit návrhové veličiny tak, aby pravděpodobnost poruchy dosáhla předem stanovené přijatelné hodnoty p_0. Zcela obecně pak lze podmínku spolehlivosti formulovat jako:

$$p_f \leq p_0 \tag{39}$$

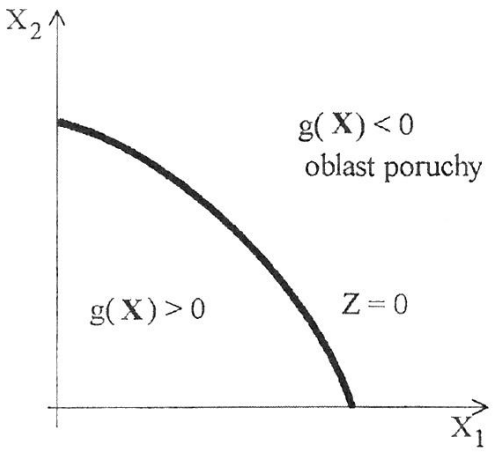
Při dalším zobecnění výpočtu pravděpodobnosti poruchy lze formulovat následující integrální vztah:

$$p_f = \int_{D_f} f(X_1, X_2, \ldots, X_n)dX_1, dX_2, \ldots, dX_n \tag{40}$$

kde D_f symbolizuje oblast poruchy tedy oblast, kde $g(\boldsymbol{X})<0$ a $f(X_1, X_2, \ldots, X_n)$ funkci sdružené hustoty pravděpodobnosti náhodných veličin $\boldsymbol{X} = X_1, X_2, \ldots, X_n$ [4]. Také řešení integrálu (40)

je ve většině případů nemožné v uzavřené formě a provádí se většinou numericky pomocí některé ze simulačních metod, jež jsou popsány v kapitole 1.3.

Zvláštní význam pro výpočet pravděpodobnosti poruchy (za užití metod aproximačních) má rovnost Z=0, jež rozděluje návrhový prostor náhodných veličin $\boldsymbol{X} = X_1, X_2, \ldots, X_n$ na oblast bezpečnou a oblast poruchy. Dvourozměrný případ definované hranice poruchy je znázorněn na Obr. 14.



Obr. 14 Znázornění oblasti poruchy ve dvourozměrném prostoru – převzato z [1]

1.5.3 INDEX SPOLEHLIVOSTI

V praxi nabývá pravděpodobnost poruchy hodnot $p_f = 10^{-q}$, kde q se pohybuje v rámci intervalu 0 až 12. Při popisu spolehlivosti pomocí p_f tedy často pracujeme s velice malými čísly, což způsobuje komplikace při práci z hlediska možného nárůstu zaokrouhlovacích chyb. Z tohoto důvodů se pro kvantifikaci spolehlivosti využívá kromě pravděpodobnosti poruchy také index spolehlivosti, který umožňuje vyjádřit spolehlivost pomocí čísla z intervalu 1 až 7. Je však třeba rozlišovat mezi elementárním indexem spolehlivosti dle Cornella a indexem spolehlivosti dle Hasofera a Linda [4]. Každý z těchto indexů je definován jinak a odpovídají si pouze v jednoduchých a specifických případech.

Cornellův elementární index spolehlivosti lze snadno demonstrovat pomocí funkce poruchy $Z = R - E$. Uvažujme R a E jako nezávislé náhodné veličiny s normálním rozdělením pravděpodobnosti. Rezerva spolehlivosti Z vypočtená dle (32) má při takto definovaných vstupních veličinách také normální rozdělení, neboť vztah (32) představuje jednoduchý součet (resp. rozdíl) dvou veličin s normálním rozdělením. Všechny popisované náhodné veličiny pak mohou být plně reprezentovány pouze svými středními hodnotami a směrodatnými odchylkami. První dva statistické momenty rezervy spolehlivosti lze vyčíslit jako:

$$\mu_Z = \mu_R - \mu_R \tag{41}$$

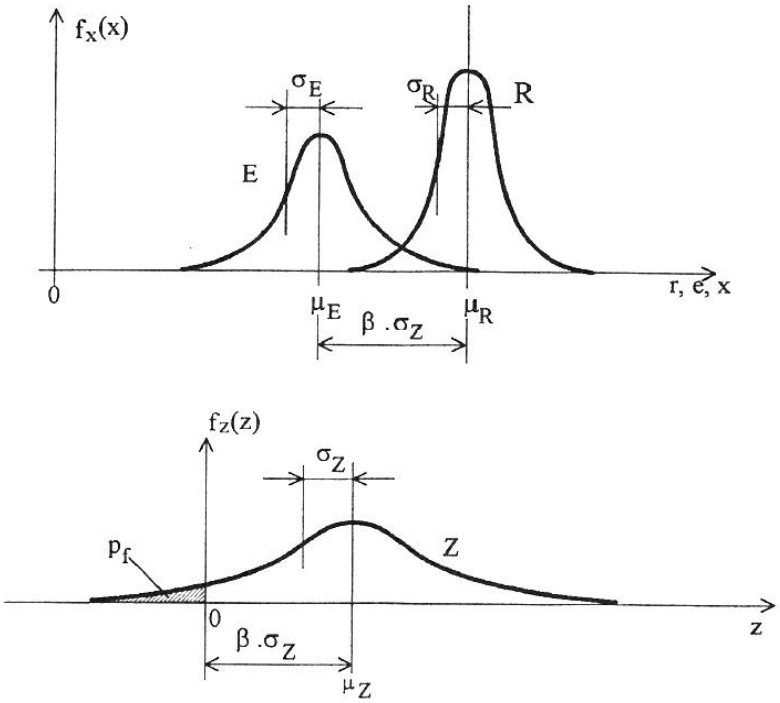
$$\sigma_Z^2 = \sigma_R^2 + \sigma_E^2 \tag{42}$$

Index spolehlivosti je pak definován vztahem:

$$\beta = \frac{\mu_Z}{\sigma_Z} \tag{43}$$

Elementární index spolehlivosti je tedy definován jako převrácená hodnota variačního koeficientu rezervy spolehlivosti *Z*. Vizuální reprezentace indexu spolehlivosti je vidět na Obr. 15. Index spolehlivosti v podstatě popisuje, kolikrát lze umístit směrodatnou odchylku rezervy spolehlivosti mezi 0 a střední hodnotu μ_Z. Pravděpodobnost poruchy lze pomocí Cornellova indexu spolehlivosti určit jako hodnotu distribuční funkce normálního rozdělení pravděpodobnosti:

$$p_f = \Phi_N(-\beta) \tag{44}$$



Obr. 15 Rezerva spolehlivosti, pravděpodobnost poruchy a elementární index spolehlivosti – převzato z [1]

Pokud je rozdělení funkce rezervy spolehlivosti výrazně odlišné od normálního, není Cornellův index spolehlivosti vhodným měřítkem pravděpodobnosti poruchy. Jeho užitím se v takovém případě můžeme dopustit hrubé chyby.

1.5.4 VÝPOČET PRAVDĚPODOBNOSTI PORUCHY POMOCÍ SIMULAČNÍCH METOD

Princip výpočtu pravděpodobnosti poruchy s použitím simulačních metod spočívá v opakovaném řešení funkce poruchy $g(X)$, pokaždé s jiným náhodně vygenerovaným vektorem vstupních náhodných veličin $\boldsymbol{X}$. V rámci j-té simulace, kde $j=1, 2, ..., N$ (N je celkový počet simulací) se pak obecně uplatňuje následující postup:

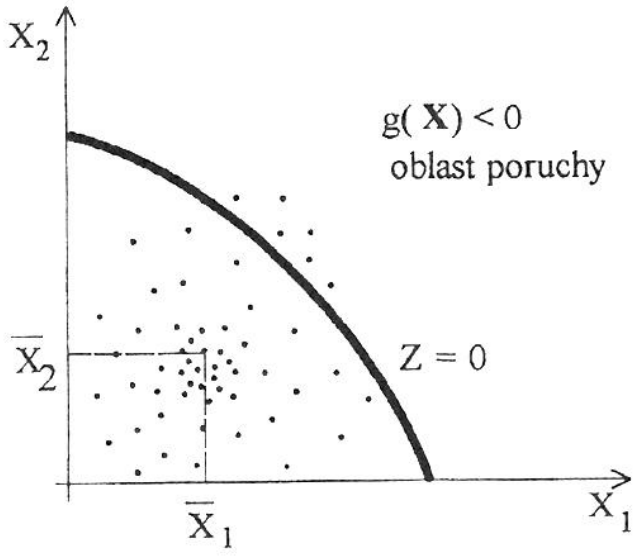
- Pomocí některé ze simulačních metod (viz kapitola 1.3) je vygenerována j-tá realizace náhodného vektoru $\boldsymbol{X} = (x_{1j}, x_{2j}, ..., x_{nj})$ v závislosti na použité simulační metodě a příslušných rozděleních pravděpodobnosti.
- V j-tém vygenerovaném bodě návrhového prostoru je pak vyčíslena hodnota funkce poruchy $g(X_1, X_2, ..., X_n)$ a odpovídající hodnota rezervy spolehlivosti pro j-tou simulaci:

$$z_j = g(x_{1j}, x_{2j}, \dots., x_{nj}) \tag{45}$$

- Po provedení všech simulací tak získáme statistický vzorek veličiny $Z (z_1, z_2, ..., z_j, ..., z_N)$. Tento pak zpracováváme postupy běžné matematické statistiky.
- Pro případ $z_j \leq 0$ dochází k poruše konstrukce. Označme celkový počet těchto případů jako N_f. Dle elementární definice pravděpodobnosti poruchy lze potom hodnotu p_f odhadnout jako poměr:

$$p_f = \frac{N_f}{N} \tag{46}$$

Obr. 16 zachycuje průběh simulace metodou Monte Carlo. Body představují jednotlivé realizace náhodného vektoru $\boldsymbol{X}$ v dvojdimenzionálním prostoru.



Obr. 16 Simulace Monte Carlo – 2D případ - převzato z [1]

Formulujme nyní výše uvedený vztah pro výpočet p_f matematicky formálněji. Zaveďme funkci $\mathbf{1}[g(\boldsymbol{X})]$, jež nabývá hodnoty 1 pro $\boldsymbol{X}$ patřící do oblasti poruchy a hodnoty 0 jinak. Integrální vztah (40) lze poté zapsat jako:

$$p_f = \int_{\Omega_x} \mathbf{1}[g(\boldsymbol{X}) < 0] f_x(\boldsymbol{X})\, d\boldsymbol{X} \tag{47}$$

Kde Ω_x je oblast integrace přes všechna $\boldsymbol{X}$. V numerickém řešení se vztah (47) nahrazuje sumací:

$$p_f = \frac{1}{N} \sum_{i=1}^{N} \mathbf{1}[g(\boldsymbol{X})] \tag{48}$$

Při užití Importance sampling je vztah (47) rozšířen o váhovou funkci $h_y(\boldsymbol{X})$, viz kapitola 1.3.3 vztah (21). Náhrada integrálu za sumaci pak probíhá dle vztahu (22).

Je evidentní, že kvalita odhadu pravděpodobnosti poruchy stoupá s rostoucím počtem simulací N. Samotná pravděpodobnost poruchy je náhodnou veličinou, její odhad pak představuje jednu z realizací této veličiny. Použijeme-li k výpočtu pravděpodobnosti poruchy prostou simulaci metodou Monte Carlo, lze pro malé pravděpodobnosti poruchy zapsat vztah pro výpočet variačního koeficientu p_f takto:

$$v_{p_f} = \frac{1}{\sqrt{N p_f}} \tag{49}$$

Ze vztahu (49) vyplývá potřeba vysokého počtu simulací potřebných ke spolehlivému odhadu velmi malých pravděpodobností poruchy.

1.6 METODY APROXIMAČNÍ

Aproximační metody byly vyvinuty za účelem redukce velkého množství simulací potřebných ke kvalitnímu odhadu malých pravděpodobností poruchy. Tyto metody můžeme rozdělit na dvě základní skupiny:

- Metody typu FORM a SORM. Tyto metody aproximují funkci poruchy jednoduchými aproximačními funkcemi, s nimiž dále pracují pomocí zdokonalených simulačních metod (např. Importance sampling). [25]

- Metody typu Response surface, jež využívají aproximace empirické distribuční funkce rezervy spolehlivosti vhodným teoretickým modelem (většinou polynomickou funkcí) [26], [27]

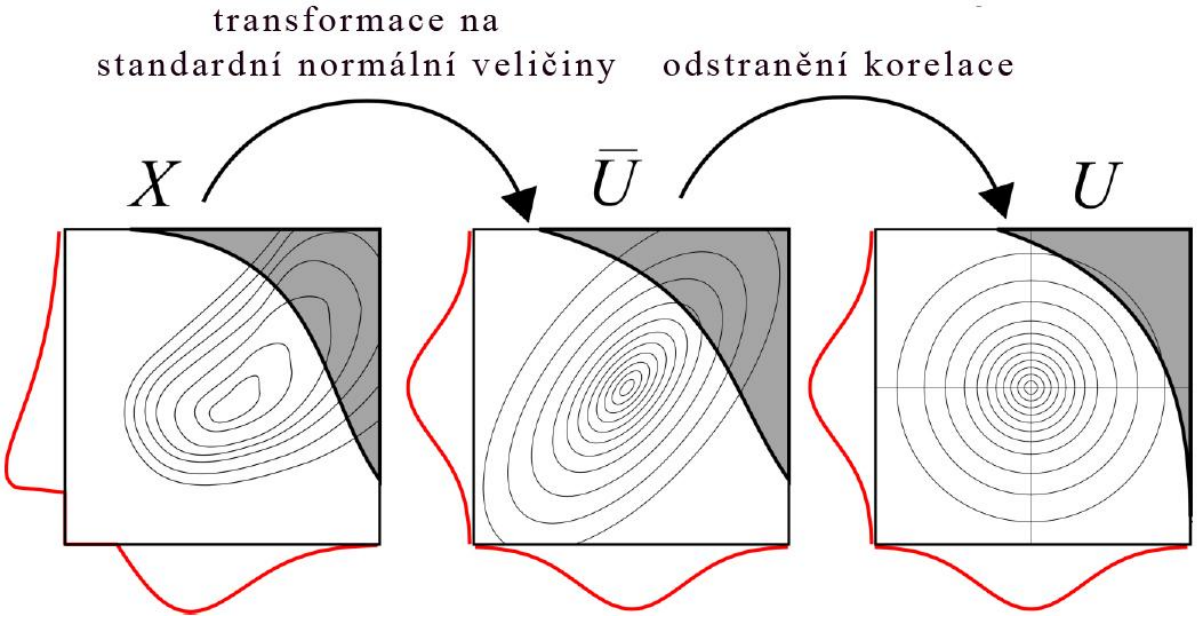
1.6.1 INDEX SPOLEHLIVOSTI DLE HASOFERA A LINDA

Hasofer-Lindův index spolehlivosti β je definován jako nejkratší vzdálenost mezi počátkem souřadnic transformované soustavy nekorelovaných normovaných normálních veličin a hranicí oblasti poruchy. Bod u^* ležící na hranici oblasti poruchy s nejkratší možnou vzdáleností od počátku soustavy transformovaných souřadnic je nazýván návrhovým bodem. Hasofer-Lindův index spolehlivosti nevyžaduje normální rozdělení pravděpodobnosti rezervy spolehlivosti. Je tedy obecnějším měřítkem spolehlivosti užívaným v rámci spolehlivostních metod druhé úrovně než Cornellův elementární index spolehlivosti.

1.6.2 FIRST ORDER RELIABILITY METHOD (FORM)

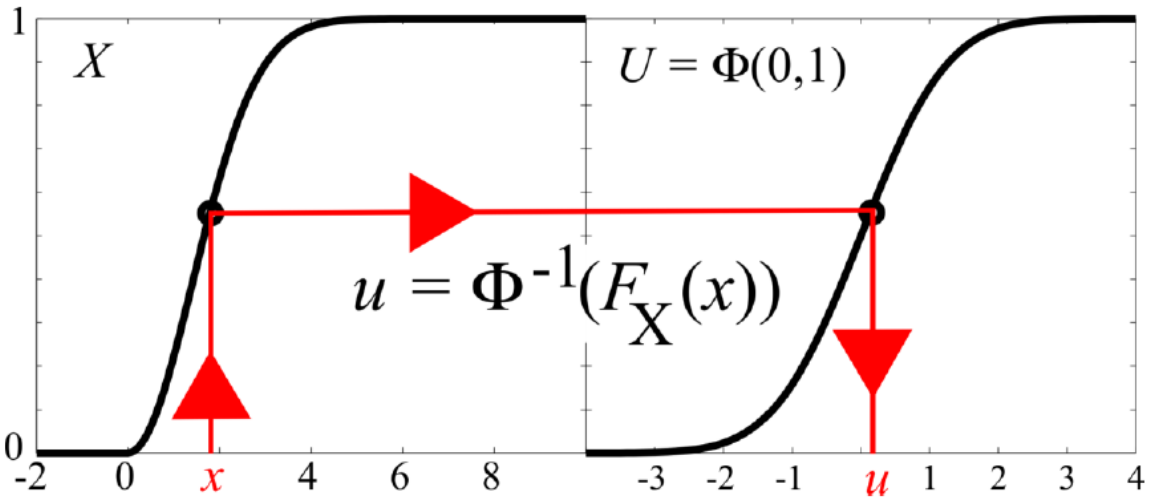
Pro výpočet Hasofer-Lindova indexu spolehlivosti metodami FORM nebo SORM je nutné transformovat všechny náhodné proměnné návrhového prostoru na standardní nekorelované normální veličiny. Metoda FORM se dá popsat v následujících krocích:

- Transformace všech veličin na standardní normální veličiny.
- Odstranění korelace.
- Hledání návrhového bodu a aproximace hranice poruchy tečnou v návrhovém bodě.



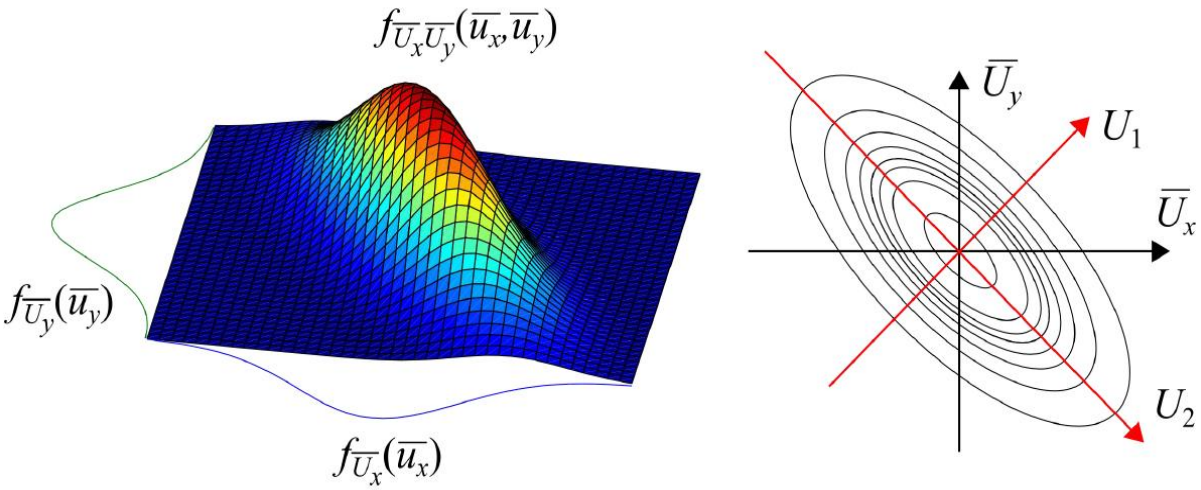
Obr. 17 Transformace návrhových veličin na standardní nekorelované normální veličiny – převzato z [54]

Existuje nekonečně mnoho bijektivních transformací, ale pouze jedna, jež je isopravděpodobnostní (zachovává si pravděpodobnost) viz Obr. 18.



Obr. 18 Isopravděpodobnostní transformace náhodné veličiny x na standardní normální veličinu u - převzato z [54]

Odstranění korelace můžeme provést například rozkladem korelační matice do vlastních vektorů. Nezávislé veličiny pak leží ve směrech vlastních vektorů (viz. Obr. 19).



Obr. 19 Rozklad korelovaných veličin na nekorelované – převzato z [54]

Následuje lokalizace návrhového bodu. Nalezení návrhového bodu může být obecně velice problematické, může existovat více lokálních extrémů (minima) vzdálenosti k počátku souřadnic. Příspěvek k výsledné pravděpodobnosti poruchy nemusí být dominantní pouze v oblasti návrhového bodu. V případech silně nelineárních funkcí mezního stavu se mohou

výsledky získané metodou FORM významně lišit od přesného řešení. Optimalizační úlohu nalezení návrhového bodu můžeme formulovat takto:

$$\beta = \sqrt{u^T u} \quad \rightarrow min \tag{50}$$

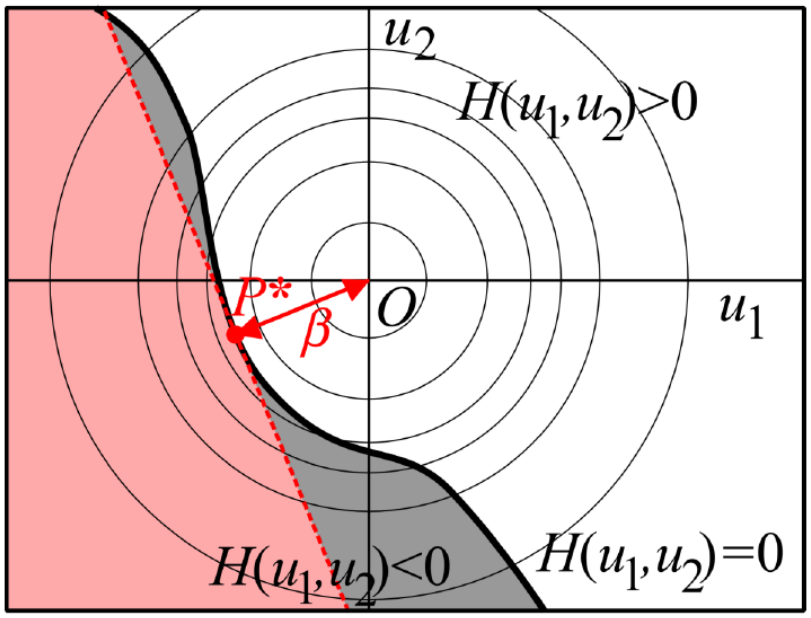
Za podmínky:

$$g(u) = 0 \tag{51}$$

Jedná se obecně o složitý mnohodimenzionální problém řešitelný nelineárními optimalizačními postupy (např. genetickými algoritmy).

Po nalezení návrhového bodu je hranice poruchy nahrazena tečnou vedoucí daným návrhovým bodem (viz. Obr. 20). Přibližná pravděpodobnost poruchy je pak vypočtena dle vztahu:

$$p_f \approx 1 - \Phi_N(-\beta) \tag{52}$$

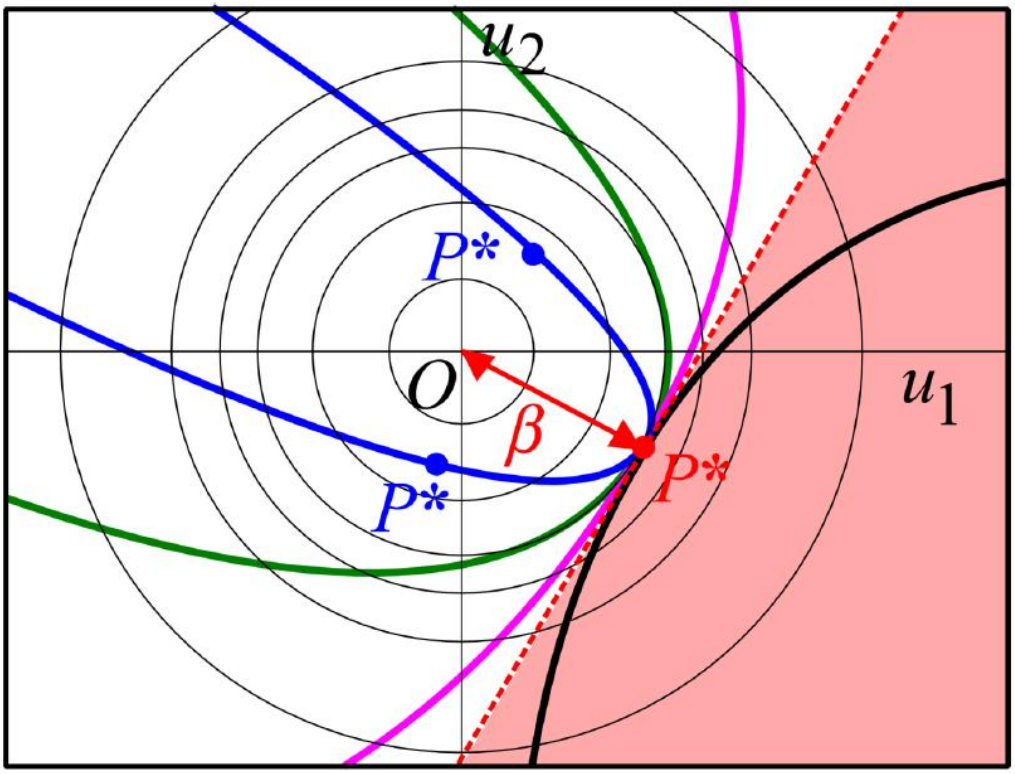


Obr. 20 Metoda FORM – aproximace hranice oblasti poruchy tečnou v návrhovém bodě - převzato z [54]

$H(u_1, u_2)$ je dvourozměrný prostor vzniklý transformací původního návrhového prostoru $G(x_1, x_2)$.

1.6.3 SECOND ORDER RELIABILITY METHOD (SORM)

Je zpřesněním metody FORM. Postup určení návrhového bodu zůstává totožný jako u metody FORM. Rozdíl spočívá v aproximaci hranice oblasti poruchy kvadratickou funkcí.



Obr. 21 Metoda SORM – aproximace hranice oblasti poruchy kvadratickou funkcí v návrhovém bodě – převzato z [54]

Metodu SORM lze považovat za mnohem přesnější než metodu FORM. Problémy spojené s určením návrhového bodu však přetrvávají.

1.6.4 METODY RESPONSE SURFACE

Přesnost výpočtu pravděpodobnosti poruchy s využitím simulačních technik je silně závislá na počtu provedených simulací. Vzhledem ke složitosti modelů pro výpočet odezvy konstrukce, jež jsou často řešeny s užitím výpočetně náročné metody konečných prvků, je provádění velkého množství simulací potřebných k určení pravděpodobnosti poruchy pomocí simulačních metod nemyslitelné. V těchto případech je vhodné použít metody Response surface. Tyto metody využívají aproximace hranice oblasti poruchy $g(\boldsymbol{X})$ vhodnou polynomickou funkcí $g_{ap}(\boldsymbol{X})$. Původní funkce mezního stavu je tak vyčíslována jen několikrát. Dobře použitelná aproximační funkce může být např. tvaru:

$$g_{ap}(\boldsymbol{X}) = a + \sum_{i=1}^{n} b_i X_i + \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij} X_i X_j \qquad (53)$$

Interpolační body pro získání koeficientů polynomu musí být vhodně zvoleny. Aproximace by měla probíhat v prostoru blízkém hranici oblasti poruchy, kde funkce poruchy mění znaménko. Aproximační polynom lze získat řešením soustavy $(n+1)(n+2)/2$ lineárních

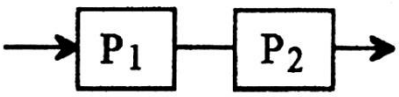
rovnic. Po získání aproximačního polynomu lze aplikovat běžné simulační postupy k získání pravděpodobnosti poruchy. Místo původní funkce je pak vyčíslován pouze jednoduchý aproximační polynom. Tímto způsobem lze analyzovat i rozsáhlé, výpočetně náročné problémy. Metody Response surface jsou dnes běžně používány hned v několika variantách blíže např. v [25].

1.7 SYSTÉMOVÁ SPOLEHLIVOST

V praxi je možné vyšetřovat spolehlivost na více úrovních. Při globálním pohledu na danou konstrukci vyšetřujeme celkovou spolehlivost systému. Mnohdy je však velice obtížné vytvořit model postihující spolehlivost systému jako celku. Podíváme-li se na konstrukci jako na soubor vzájemně propojených konstrukčních prvků, z nichž každý má svou vlastní kvantifikovatelnou spolehlivost, můžeme s ohledem na vzájemné vazby mezi prvky a jejich role v rámci systému jako celku stanovit celkovou spolehlivost systému na základě spolehlivostí jednotlivých prvků. V souvislosti s rozšířenou metodikou navrhování podle mezních stavů je druhý zmíněný přístup běžnější. Následující text popisuje pouze základní principy rozsáhlé problematiky systémové spolehlivosti.

1.7.1 SYSTÉMY SÉRIOVÉ

V sériovém systému jsou jednotlivé prvky zapojeny tak, že selhání jediného z nich vede ke kolapsu systému jako celku. Ideálním příkladem sériového systému je řetěz. Porušením jednoho z jeho článků selhává i řetěz jako konstrukční prvek. Konstrukce staticky určité jsou rovněž sériovými systémy, neboť selháním jednoho jejich prvku dochází k selhání systému jako celku (např. selhání některého z prutů staticky určité příhradové konstrukce). Schéma sériového zapojení prvků je na Obr. 22.



Obr. 22 Sériové zapojení prvků systému – převzato z [1]

Za předpokladu statistické nezávislosti selhání jednotlivých prvků je spolehlivost sériového systému dána vztahem:

$$R = (1 - p_{f1})(1 - p_{f2}) \dots . (1 - p_{fn}) = \prod_{i=1}^{n} (1 - p_{fi}) \tag{54}$$

Pravděpodobnost poruchy systému je pak doplněk spolehlivosti do jedničky:

$$P_f = 1 - \prod_{i=1}^{n}(1 - p_{fi}) \approx \sum_{i=1}^{n} p_{fi} \tag{55}$$

Přibližná sumace platí pro malé p_{fi}. Dle (55) pravděpodobnost poruchy roste se zvýšením počtu prvků systému a je silně závislá na pravděpodobnosti poruchy nejméně spolehlivého prvku.

1.7.2 SYSTÉMY PARALELNÍ

Systémy paralelní jsou takové, ve kterých jsou jednotlivé prvky zapojeny paralelně, tedy tak, že k selhání sytému jako celku je potřeba, aby selhaly všechny zapojené prvky. Za předpokladu statistické nezávislosti selhání jednotlivých prvků je pravděpodobnost selhání paralelního systému dána vztahem:

$$P_f = p_{f1} \times p_{f2} \times \ldots \times p_{fn} = \prod_{i=1}^{n} p_{fi} \tag{56}$$

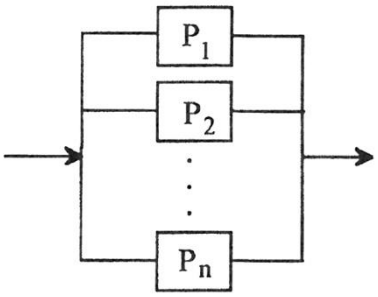
V případě dokonalé statistické závislosti platí:

$$P_f = min\{p_{fi}\} \tag{57}$$

Pravděpodobnost poruchy sytému tedy nesmí překročit pravděpodobnost poruchy nejméně spolehlivého prvku. Z uvedeného vyplývá, že pravděpodobnost poruchy paralelního systému musí být v rámci intervalu:

$$\prod_{i=1}^{n} p_{fi} \leq P_f \leq min\{p_{fi}\} \tag{58}$$

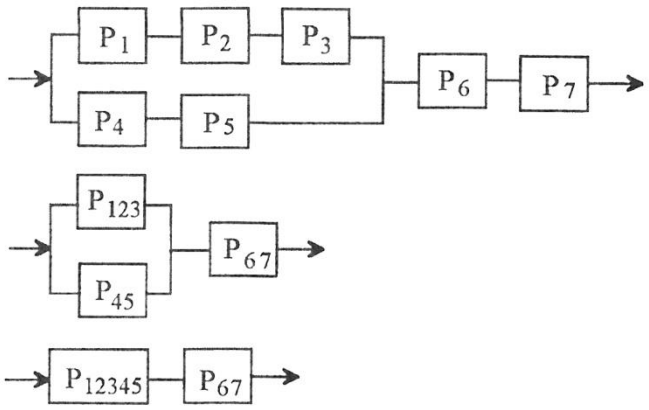
Paralelní systémy jsou tedy spolehlivější než sériové. Mezi paralelní konstrukční systémy se teoreticky řadí staticky neurčité konstrukce. Ve stavební praxi ovšem zřídkakdy narazíme na systémy pracující čistě v paralelním smyslu.



Obr. 23 Paralelní zapojení systémových prvků – převzato z [1]

1.7.3 SYSTÉMY KOMBINOVANÉ

V praxi nejčastěji přítomné systémy využívají kombinace sériových i paralelních zapojení. Takovéto systémy řešíme postupnou redukcí paralelních či sériových prvků na systémy jednodušší. Tento postup je zachycen na Obr. 24.



Obr. 24 Kombinace sériového a paralelního zapojení prvků – převzato z [1]

1.8 ANALÝZA CITLIVOSTI (SENSITIVITY)

Citlivostní analýza se zabývá určením vlivu vstupních veličin na funkční hodnotu zkoumané funkce. Matematický model je různě citlivý na změnu jednotlivých vstupních parametrů. Například moment na mezi únosnosti v ohybu nosníku obdélníkového průřezu je závislý na velikosti průřezového modulu. Ve vzorci pro výpočet průřezového modulu $\left(W = \frac{1}{6}bh^2\right)$ je hodnota výšky h umocněna zatímco šířka průřezu b ne. Je zřejmé, že změnou hodnoty h ovlivníme výslednou hodnotu momentu na mezi únosnosti více než změnou hodnoty b. Uvedená funkce je tedy citlivější na změnu hodnoty h. Popsaný příklad je velice jednoduchý a větší citlivost funkční hodnoty na změnu hodnoty parametru vyššího řádu je zřejmá na první pohled. V praxi se však setkáváme se složitými modely (často pracujícími na bázi metody konečných prvků), u nichž je určení citlivosti vůči změnám vstupních veličin obtížné. U náhodných veličin nás většinou zajímá, jakým způsobem ovlivňují rozptyl např. odezvy konstrukce a jak se podílí na výsledné pravděpodobnosti poruchy. Citlivostní analýza nám tedy poskytuje informace o významu jednotlivých vstupních parametrů. Na jejím základě můžeme například určit, které z parametrů významně ovlivní pravděpodobnost poruchy a které naopak nemají na p_f velký vliv a můžeme je tudíž definovat jen deterministicky, aniž bychom se dopustili významné chyby.

Deterministická analýza sensitivity je známým běžně užívaným prostředkem při návrhu konstrukcí. Většinou je realizována formou tzv. parametrické studie. Taková studie může být organizována jako posloupnost výpočtů s postupně se měnící hodnotou některého ze vstupních parametrů v každém z výpočtových kroků v rámci určitého reálného rozsahu. Vliv parametru na funkční hodnotu zájmové funkce pak můžeme určit srovnáním výsledků

jednotlivých výpočtových kroků. Citlivostní analýza je také často vedlejším produktem deterministických optimalizačních postupů pracujících s gradienty cílové funkce. Základní charakteristikou deterministické citlivostní analýzy je deterministická (nenáhodná) povaha uvažovaných vstupních parametrů.

Stochastická analýza sensitivity poskytuje komplexnější informace o vlivu jednotlivých vstupních parametrů. Používá náročnějších numerických metod. Vstupní proměnné jsou zde uvažovány jako náhodné veličiny s příslušným rozdělením pravděpodobnosti s danými statistickými parametry (směrodatná odchylka, střední hodnota, koeficient šikmosti a špičatosti). V nedávné době byla rozvinuta celá řada různých metod stochastické citlivostní analýzy (souhrn [55]). Užívané metody je možno rozdělit na dvě skupiny podle zjišťování vlivu vstupních parametrů na:

- pravděpodobnost poruchy (failure sensitivity – FPS)
- odezvu konstrukce (response sensitivity – RS)

Přístupy spadající do druhé skupiny jsou založeny většinou na standardní simulaci Monte Carlo či lépe LHS. Jsou lehce implementovatelné do kteréhokoliv simulačního programu.

NEPARAMETRICKÁ POŘADOVÁ KORELACE

K určení relativního vlivu vstupní náhodné veličiny na odezvu konstrukce je možné využít hodnotu dílčího korelačního koeficientu mezi vstupní náhodnou veličinou a veličinou odezvy konstrukce. Existuje předpoklad, že veličina, jež ovlivňuje odezvu konstrukce nejvíce (ať už v pozitivním či negativním smyslu), bude mít hodnotu zmíněného dílčího korelačního koeficientu vyšší než vstupní náhodné veličiny ovlivňující odezvu konstrukce méně.

Při tomto způsobu stanovení citlivosti veličiny odezvy na změnu dané vstupní náhodné veličiny je možné využít jednoduché metody Monte Carlo, dokonalejší metody stratifikované simulace (např. metody LHS) jsou však výhodnější. Metoda využívá náhodné permutace čísel vrstev na distribučních funkcích vstupních náhodných veličin. Pro citlivostní analýzu je výhodné použít neparametrickou pořadovou korelaci, protože analyzovaný výpočetní model je v obecném případě nelineární a klasický korelační koeficient je použitelný pouze pro veličiny normálně rozdělené.

Neparametrickou pořadovou korelaci můžeme určit pomocí následujícího postupu: Místo skutečných hodnot ve statistických souborech vstupních náhodných veličin a veličiny odezvy se pracuje s pořadím jednotlivých realizací v souborech seřazených dle skutečných hodnot zmíněných veličin, tj. *1, 2, …., N*. Výsledný soubor čísel je pak vybírán ze známého rozdělení pravděpodobnosti – celá čísla mezi 1 a N jsou rovnoměrně rozdělena.

Neparametrická korelace je ve srovnání s klasickou lineární korelací odolnější vůči defektům v datových souborech a je nezávislá na rozdělení pravděpodobnosti příslušných souborů. Pro potřeby statistické analýzy pomocí neparametrické pořadové korelace se používají následující statistiky: Spearmanův korelční koeficient (18) a Kendallovo tau. [1]

2. SOFTWAROVÉ PROSTŘEDKY

Vhodná automatizace postupu spolehlivostní optimalizace je základním předpokladem jeho úspěšné aplikace při řešení praktických úloh. Existuje celá řada programových prostředků pro optimalizaci a spolehlivostní posuzování [1]. Mnohé z algoritmů spolehlivostního posuzování jsou dnes aplikovány jako doplněk či nástavba programů pro posuzování konstrukcí metodou konečných prvků. Optimalizační algoritmy se již dříve staly součástí programů jako je např. ANSYS. Spojení optimalizace a spolehlivostního posuzování matematického modelu v rámci jediného programu je však dnes stále spíše výjimečnou záležitostí.

Cílem dále představené automatizace procesu spolehlivostní optimalizace bylo propojení optimalizačního postupu se spolehlivostním posuzováním v rámci uceleného programu umožňujícího optimalizaci modelu algoritmem AMS se zohledněním spolehlivostních omezujících podmínek. Pro účely simulace a spolehlivostního posouzení modelu byl využit dostupný software FReET.

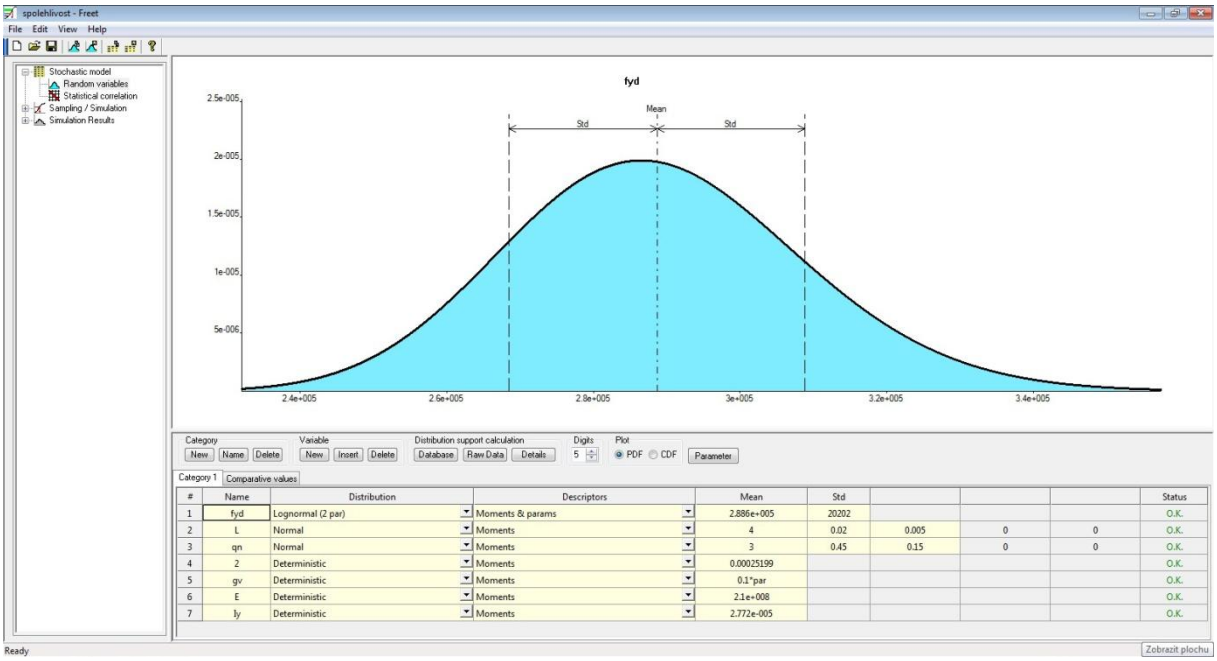
2.1 FREET

FReET (Feasible Reliability Engineering Tool) je 32-bitový software vyvinutý na Ústavu stavební mechaniky fakulty Stavební Vysokého učení technického v Brně [28]. Slouží jako víceúčelový pravděpodobnostní software určený pro statistickou analýzu, analýzu citlivosti a analýzu spolehlivosti při řešení technických problémů. Obsahuje nástroje pro modelování náhodných veličin a polí (databáze pravděpodobnostních rozdělení, editor parametrů přiděleného rozdělení atd.). Pro simulaci v rámci návrhového prostoru užívá metod Monte Carlo a LHS (mean, median, random). Umožňuje zavedení požadované korelace mezi vygenerovanými náhodnými veličinami úpravou tabulky náhodných permutací s využitím metody Simulovaného žíhání. Disponuje také nástroji pro definici cílové funkce (editor rovnic či definice v rámci DLL). Díky možnosti definovat problém v DLL je FReET schopen pracovat se širokou škálou problémů včetně nelineárních modelů využívajících metody konečných prvků. Výpočty indexu spolehlivosti a pravděpodobnosti poruchy jsou v rámci FReETu realizovány metodou FORM. Program také obsahuje základní nástroje k analýze rizik. Výhodou oproti některým jiným dostupným programům je přehledné (GUI) prostředí FReETu s možností vizualizace výsledků a vstupních rozdělení (viz Obr. 26).

Díky spolupráci s firmou Červenka Consulting vyvíjející program ATENA (Advanced Tool for Engineering Nonlinear Analysis), jež slouží k nelineární numerické analýze konstrukcí metodou konečných prvků, bylo vyvinuto softwarové prostředí SARA Studio (Structural Analysis and Reliability Assessment) umožňující propojení programů ATENA a FReET. Bylo tak umožněno randomizovat vybrané parametry modelu vytvořeného v programu ATENA pomocí nástrojů softwaru FReET. Je proto možné provádět spolehlivostní a degradační analýzy složitých nelineárních problémů či modelovat

proměnlivost materiálových vlastností pomocí náhodných polí. Prostředí SARA Studio je tedy komplexním nástrojem pro spolehlivostní analýzu složitých modelů.

Výstupy (popř. vstupy) programu FReET lze snadno ukládat do textových souborů s příponou .fre. Tyto je možné snadno automaticky editovat pomocí programátorských technik pro práci s běžnými textovými soubory [56]. Program byl proto zvolen jako vhodný výpočetní základ pro aplikaci FNPO, popsanou v následující kapitole, jež funguje jako obslužný program pro výpočetní "jádro" FReETu umožňující aplikaci simulačních metod typu Monte Carlo. Bližší informace o programu FReET jsou k dispozici v [28].



#	Name	Distribution	Descriptors	Mean	Std				Status
1	fyd	Lognormal (2 par)	Moments & params	2.886e+005	20202				O.K.
2	L	Normal	Moments	4	0.02	0.005	0	0	O.K.
3	qn	Normal	Moments	3	0.45	0.15	0	0	O.K.
4	2	Deterministic	Moments	0.00025199					O.K.
5	gv	Deterministic	Moments	0.1*par					O.K.
6	E	Deterministic	Moments	2.1e+008					O.K.
7	Iy	Deterministic	Moments	2.772e-005					O.K.

Obr. 26 Kopie obrazovky při práci s programem FReET

2.2 FREET NESTED PROBABILISTIC OPTIMIZER (FNPO)

2.2.1 POPIS PROGRAMU

FNPO je akademický software vyvinutý primárně pro účely spolehlivostní optimalizace (v rámci SSA) a testování algoritmu AMS. Program funguje jako řídící software procesu spolehlivostní optimalizace využívající program FReET, který realizuje základní výpočty jednotlivých fází algoritmu programu FNPO. Samotný FReET je vytvořen v programovacím jazyce C++. Vzhledem k nutnosti práce s některými funkcemi FReETu uloženými v dynamických knihovnách (DLL) vytvořených rovněž v C++, byl pro zápis programu FNPO zvolen také tento programovací jazyk [57]. Z široké nabídky dostupných překladačů byl pro účely práce na FNPO zvolen přehledný překladač s množstvím předprogramovaných komponent, tříd a objektů Borland C++ Builder 6 [58]. Užitý překladač je hojně používán a je k němu k dispozici celá řada podpůrných a výukových materiálů.

Aplikace FNPO funguje v prostředí operačního systému Windows a je 32 bitová stejně jako FReET.

FNPO umožňuje cyklické spouštění FReETu a vyhodnocování dat získaných v každém z cyklů, na jejichž základě je poté sestaven další vstupní soubor pro FReET (s příponou .fre) předaný FReETu v následujícím cyklu k vyhodnocení. Tento proces se opakuje, dokud není dosaženo dané podmínky ukončení iteračního procesu. Ve FNPO je tato podmínka definována limitním počtem úrovní algoritmu AMS.

Program pracuje s tzv. double loop přístupem ke spolehlivostní optimalizaci. V tomto přístupu pracuje algoritmus ve dvou (až třech) základních cyklech:

- **Vnější cyklus** představuje optimalizační část procesu. V tomto cyklu je prováděna simulace v rámci návrhového prostoru. Pro získané návrhové vektory n-rozměrného prostoru $X_i(x_1, x_2, ..., x_n)$ je pak vypočtena funkční hodnota cílové funkce. Na základě těchto hodnot je vybrána nejlepší realizace. Následně dochází ke konfrontaci nejlepší realizace $X_{i,best}$ s omezujícími podmínkami optimalizace. Tyto podmínky mohou být formulovány jakoukoli deterministickou funkcí, jejíž funkční hodnotu porovnáme s definovaným intervalem povolených hodnot. Omezující podmínku je také možné formulovat spolehlivostně. Pro jakoukoli definovanou funkci mezního stavu (v rámci návrhového prostoru úlohy) můžeme zavést interval povolených hodnot indexu spolehlivosti β. Výpočet indexu spolehlivosti jednotlivých vygenerovaných náhodných vektorů X_i probíhá ve vnitřní smyčce. Vyhoví-li náhodný vektor omezujícím podmínkám, je přijat jako další výchozí bod algoritmu AMS nebo v případě optimalizace prostou simulací jako hledané řešení.

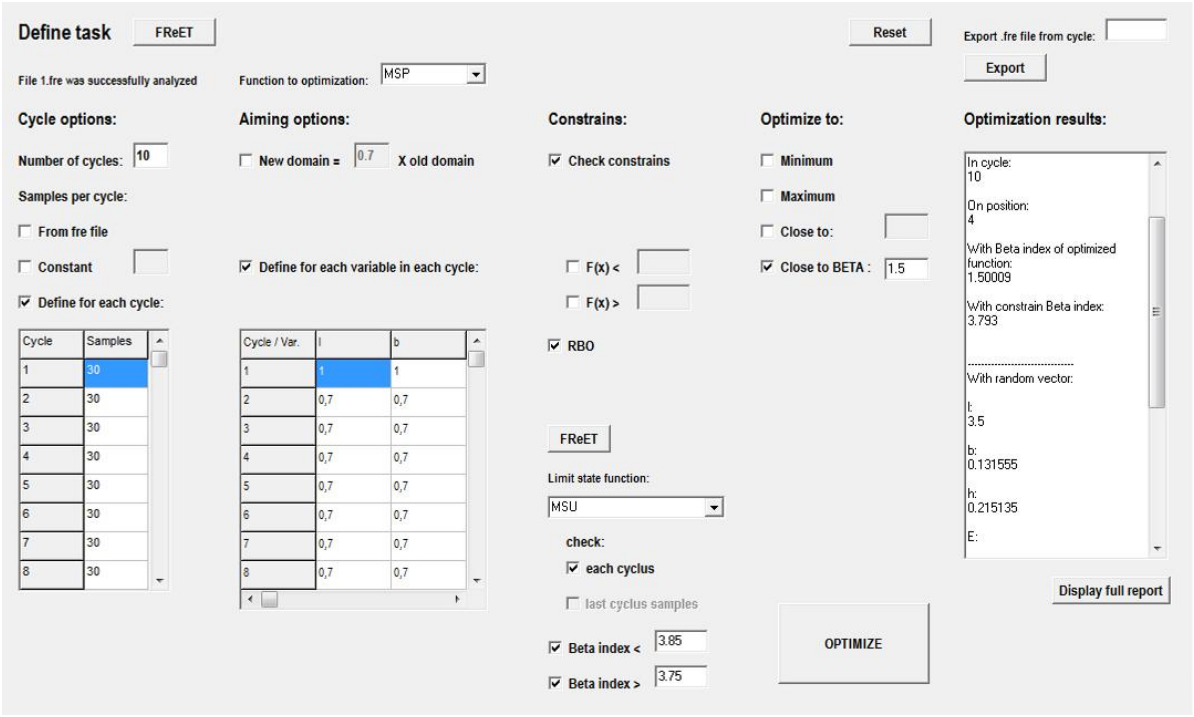
  Program FNPO umožňuje také jiný přístup ke spolehlivostní optimalizaci. Samotná cílová funkce může mít funkční hodnoty v podobě indexu spolehlivosti. FNPO umožňuje definovat cílovou hodnotu indexu spolehlivosti pro jakoukoli v daném návrhovém prostoru definovanou funkci mezního stavu. Zároveň je možné definovat spolehlivostní či deterministické omezující podmínky. Při této definici se k hlavnímu cyklu optimalizace a vnořenému cyklu pro výpočet spolehlivosti k porovnání s omezujícími podmínkami přidává ještě třetí vnořený cyklus pro výpočet indexu spolehlivosti jako funkční hodnoty cílové funkce.

- **Vnitřní cyklus/cykly** slouží k výpočtu indexu spolehlivosti buď pro potřebu konfrontace generovaného řešení s omezujícími podmínkami, nebo pro výpočet samotné funkční hodnoty cílové funkce optimalizace, je-li index spolehlivosti cílovou veličinou optimalizačního procesu.

FNPO není samostatně použitelným programem. Ke své práci potřebuje software FReET, v němž probíhá definice všech funkcí, proměnných, příslušných rozdělení pravděpodobnosti a korelační matice. Vnitřní cykly pro výpočet indexů spolehlivosti jsou také

realizovány v rámci FReETu aproximační metodou FORM. FNPO pak vhodným způsobem zpracovává .fre soubory a řídí souběh všech úrovní procesu spolehlivostní optimalizace. Program je tedy plně závislý na výpočtech FReETu a není možné jej používat samostatně, a to jak pro deterministickou tak pro spolehlivostní optimalizaci.

Pro samotný proces optimalizace nabízí FNPO použití dvou metod. Uživatel má na výběr, zda využije prosté simulace některou ze simulačních metod dostupných ve FReETu nebo algoritmu AMS, jehož testování bylo jednou z motivací k vývoji programu FNPO. Algoritmus AMS je možné plně kontrolovat a v programu jsou implementovány všechny možnosti jeho nastavení popsané v kapitole 1.4.3. Vzhled pracovního prostředí programu je zachycen na Obr. 27.



Obr. 27 Kopie obrazovky při práci s aplikací FNPO

Uživatelské prostředí je koncipováno do 5 základních sloupcově uspořádaných sekcí:

- Cycle options
- Aiming options
- Constrains
- Optimize to
- Optimization results

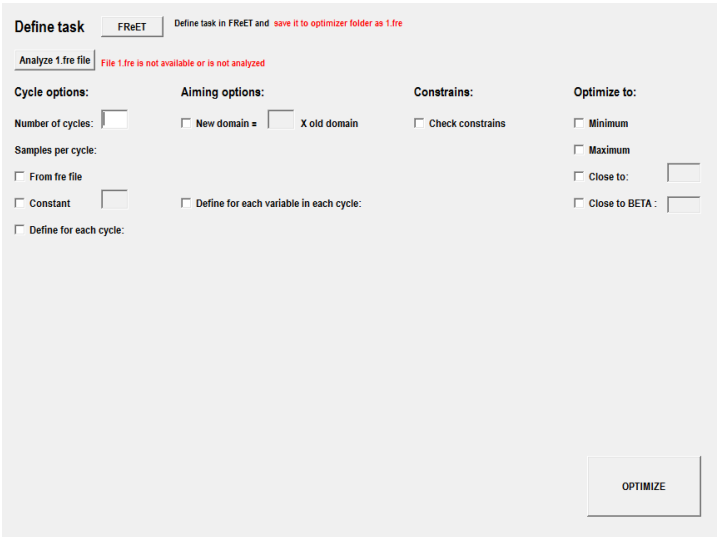
Podrobný popis možných nastavení jednotlivých sekcí spolu s návodem k obsluze programu je k dispozici v uživatelském manuálu, který je součástí příloh (P1).

Jedním z důvodů vývoje FNPO bylo vytvořit prostředek pro testování účinnosti algoritmu AMS. K tomuto účelu je nezbytné sledovat průběh algoritmu na všech jeho úrovních. Program FNPO proto ukládá kompletní záznamy optimalizačního procesu včetně informací o aplikaci daných omezení a úspěšnosti přijetí náhodných vektorů v závislosti na omezujících podmínkách optimalizace. Je také možné exportovat kterýkoli z .fre souborů užívaných během procesu spolehlivostní optimalizace včetně souborů původních. Textové soubory obsahující záznamy o průběhu procesu rovněž obsahují informace o všech vypočtených indexech spolehlivosti ve kterémkoli z pracovních cyklů. Zprávy o průběhu spolehlivostní optimalizace jsou z důvodu přehlednosti rozděleny do několika textových souborů:

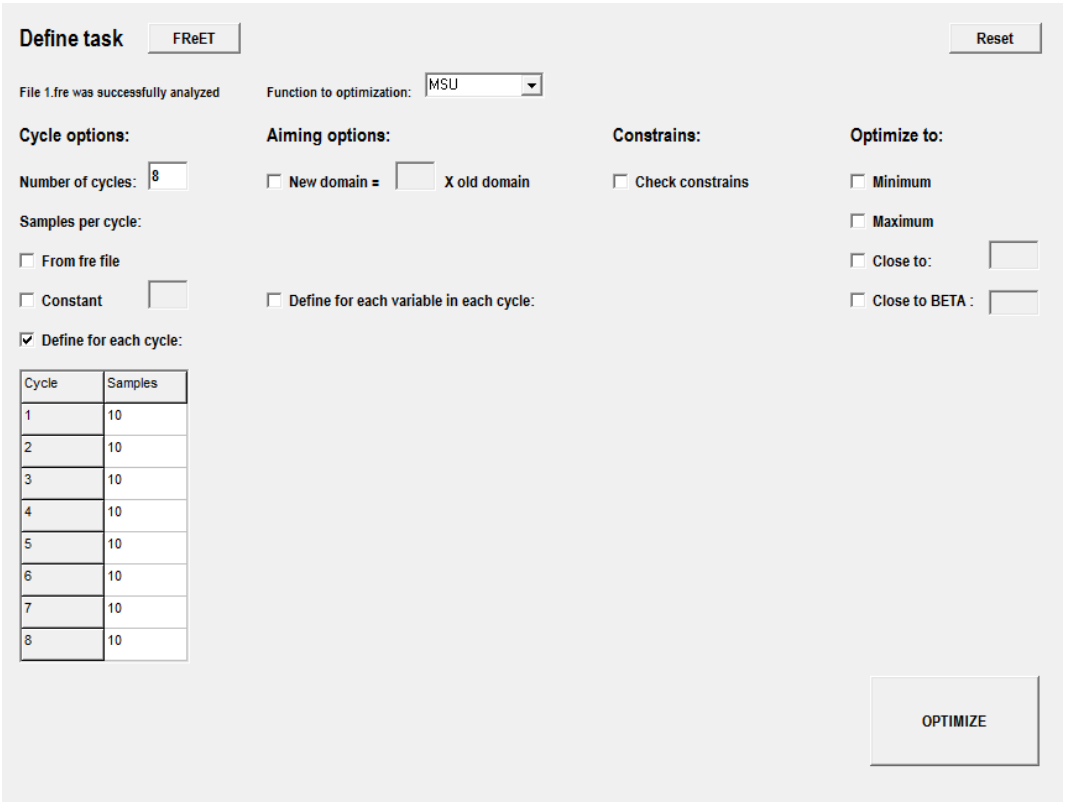
- Optimization report
- Analysis report
- Analysis 2 report
- RBO list report
- RBO results

Přehled obsahu a struktura těchto zpráv jsou popsány v přiloženém uživatelském manuálu (P1).

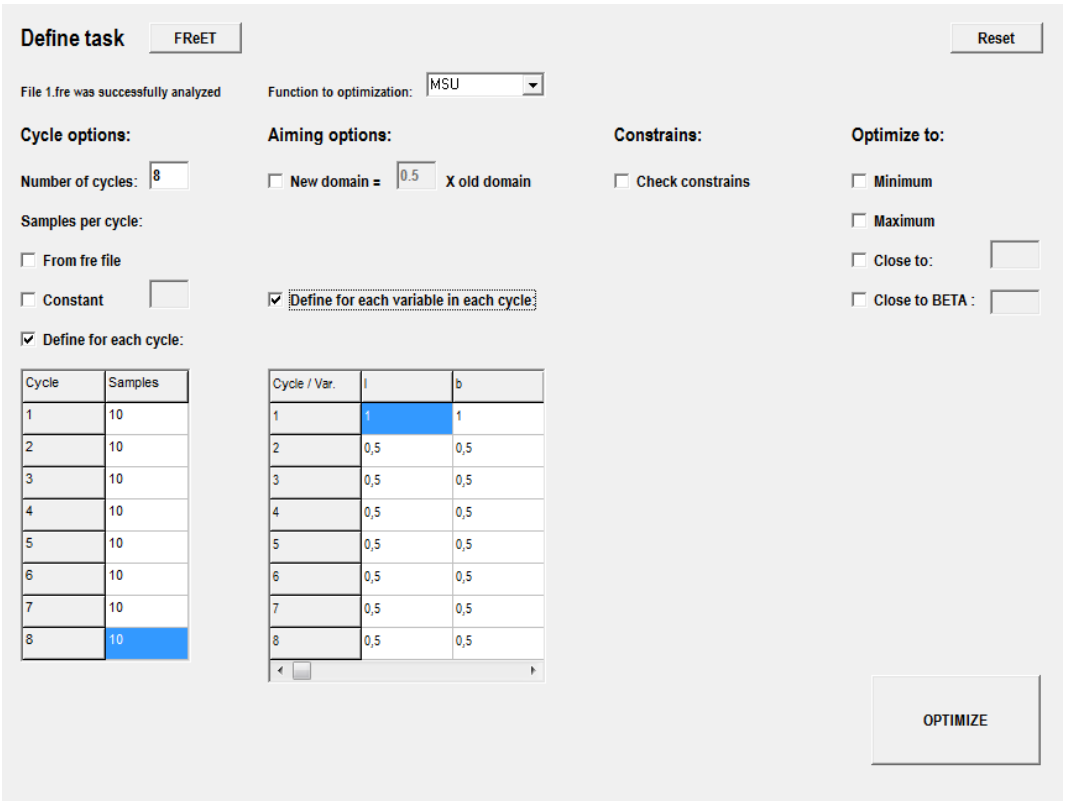
Vzhledu uživatelského prostředí a přehlednosti aplikace nebyla doposud věnována velká pozornost. Prozatím je tedy program určen jen pro uživatele s hlubší znalostí problematiky spolehlivostní optimalizace. FNPO však již nyní poskytuje celou řadu kontrol proti chybnému zadání parametrů procesu. Uživatel je provázen celým nastavením procesu optimalizace krok po kroku a další parametry nastavení jsou zpřístupněny teprve po definici podstatných parametrů předchozích sekcí. Program je také chráněn proti chybnému zadávání numerických parametrů. Postupné zpřístupňování jednotlivých sekcí programu je vidět na Obr. 28-32.



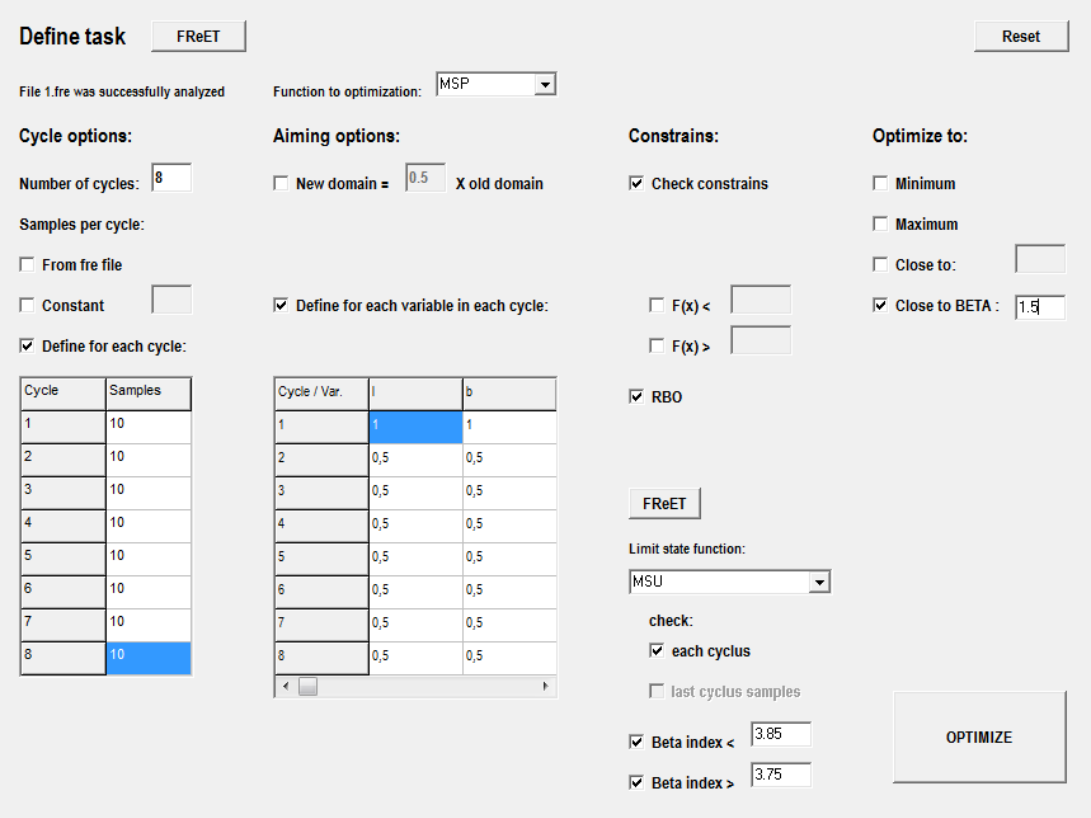
Obr. 28 FNPO po spuštění



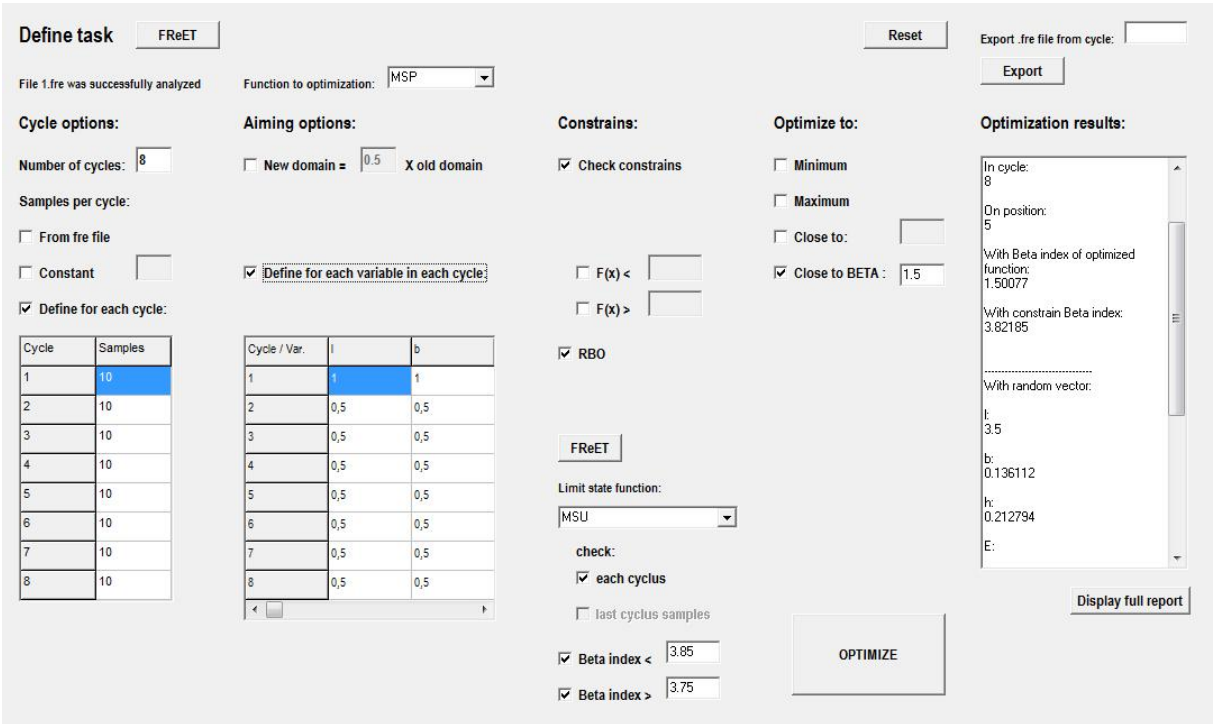
Obr. 29 FNPO po načtení .fre souboru a definice počtu cyklů a vzorků



Obr. 30 Nastavení "cílení" algoritmu AMS



Obr. 31 Nastavení omezujících podmínek a cíle optimalizace



Obr. 32 Zobrazení výsledků po provedení optimalizace

Program také umožňuje pokračovat dále v optimalizačním procesu algoritmem AMS od bodu, kde optimalizace skončila. Pokud tedy není uživatel s výsledkem spokojen, je možné jej nadále zpřesňovat.

2.2.2 PŘÍKLADY PROBLÉMŮ ŘEŠITELNÝCH POMOCÍ FNPO

Pomocí programu FNPO lze řešit celou řadu uživatelem definovaných optimalizačních problémů. Popsaná verze programu je však zatím stále ve stadiu testování. Některé funkce, s nimiž se do budoucna počítá, doposud nebyly plně implementovány. Program tak zatím trpí několika omezeními. Řešené problémy v současnosti nemohou být multikriterální a existuje zde možnost zadat pouze jednu funkci jako omezující podmínku (deterministickou nebo spolehlivostní). Tyto a mnohé další drobné nedostatky (týkající se GUI prostředí) by měly být odstraněny v následujících verzích programu. V dalším textu se proto zaměříme pouze na definice optimalizačních problémů, jež jsou s užitím FNPO řešitelné již dnes.

MOŽNÉ DEFINICE CÍLE OPTIMALIZACE

Nejjednodušším typem úloh řešitelných pomocí FNPO je prostá neohraničená optimalizace cílové funkce buďto s využitím prosté simulace některou ze simulačních metod implementovaných ve FReETu, nebo pomocí algoritmu AMS. Takto definované optimalizační problémy slouží hlavně pro testování účinnosti algoritmu AMS (viz kapitola 3). Přiklad neohraničené minimalizace cílové funkce může být definován:

$$f(\boldsymbol{X}) \rightarrow \min \qquad \boldsymbol{X} \in R_n \tag{59}$$

kde $\boldsymbol{X}$ je náhodný vektor v rámci definovaného n rozměrného návrhového prostoru R_n. Takto definovaná optimalizační úloha může být použita při řešení rovnic, jež nejsou řešitelné analyticky v uzavřeném tvaru. Tento typ rovnic se například objevuje v matematických modelech lanového chování. Příklad využití programu FNPO pro tyto úlohy je popsán v kapitole 4.1.

FNPO také umožňuje hledat takové náhodné vektory $\boldsymbol{X}$ v rámci definovaného návrhového prostoru R_n, jímž odpovídá definovaná funkční hodnota cílové funkce $f(\boldsymbol{X})$.

$$|f(\boldsymbol{X}) - k| \rightarrow \min \qquad \boldsymbol{X} \in R_n \tag{60}$$

kde k je definovaná funkční hodnota, pro niž se hledá vektor vstupních hodnot. Úloha je tedy definována jako minimalizace absolutní hodnoty rozdílu mezi funkční hodnotou a definovanou hodnotou k. Tento typ optimalizace je často součástí návrhu lanových konstrukcí. Architekt definuje tvar navrhované lanové konstrukce. Projektant poté musí najít takovou kombinaci zatížení, průřezové plochy a předpětí lan, aby se výsledný průhyb jednotlivých lan co nejvíce přiblížil průhybu navrženému.

Další možností je definovat cílový index spolehlivosti β_d pro danou funkci limitního stavu. Pro vygenerované náhodné vektory je tak kromě funkční hodnoty cílové funkce vypočten i index spolehlivosti, který je kritériem výběru nejlepší realizace v daném cyklu. Výpočet indexu spolehlivosti probíhá ve vnitřním cyklu programu FReET.

$$|\beta(\boldsymbol{X}) - \beta_d| \rightarrow \min \qquad \boldsymbol{X} \in R_n \tag{61}$$

Tímto způsobem zadání lze algoritmus AMS směřovat k řešení s přesně danou hodnotou spolehlivosti.

MOŽNÉ DEFINICE OMEZUJÍCÍ PODMÍNKY

Pro výše specifikované cíle optimalizace je možné současně definovat omezující podmínku ve formě omezení funkční hodnoty jedné vybrané funkce (může být omezena i funkční hodnota cílové funkce). Omezení funkční hodnoty lze aplikovat formou definice otevřeného či uzavřeného intervalu hodnot. Omezující funkce může být obecně jakákoli funkce definovaná v R_n. Příklad optimalizace s omezením:

$$f(\boldsymbol{X}) \rightarrow \min \qquad \boldsymbol{X} \in R_n \tag{62}$$

Omezeno:

$$d < c(\boldsymbol{X}) < h \tag{63}$$

nebo

$$c(\boldsymbol{X}) < h \tag{64}$$

nebo

$$c(\boldsymbol{X}) > d \tag{65}$$

kde d je dolní mez funkční hodnoty a h horní.

Omezující podmínka může být také formulována spolehlivostně. Podobně jako u deterministického omezení lze definovat otevřený nebo uzavřený interval povolených hodnot indexu spolehlivosti u dané funkce mezního stavu.

$$d < \beta_o(\boldsymbol{X}) < h \tag{66}$$

Definice otevřených intervalů jsou pro spolehlivostní omezení obdobou vztahů (64) a (65).

Funkcí mezního stavu můžeme definovat i více (např. pro mezní stav únosnosti a mezní stav použitelnosti). V takovém případě lze úlohu definovat jako optimalizaci indexu spolehlivosti jedné funkce limitního stavu za předpokladu omezení povolených indexů spolehlivosti druhé funkce limitního stavu. Tento postup je názorně demonstrován na příkladu uvedeném v kapitole 4.3. Taková definice by pak odpovídala vztahu (61) za omezení daného vztahem (66).

3. TESTOVÁNÍ ALGORITMU AMS

Optimalizační problémy objevující se napříč vědními a průmyslovými obory jsou většinou mnohodimensionální. U problémů mnoha oborů (např. logistika, skladování, ekonomika, informatika) řešených moderními optimalizačními algoritmy není vyčíslení funkční hodnoty cílové funkce náročné. Je proto důležitější přesná lokalizace extrému než rozumný počet simulací potřebný k nalezení přijatelného řešení. Optimalizační úlohy běžné ve stavební praxi jsou v tomto ohledu odlišné. Z důvodu často velké výpočetní náročnosti funkční hodnoty cílové funkce je možné pracovat s maximálně stovkami simulací. Počet dimenzí pro řešení spolehlivostních úloh je většinou vysoký, naopak v optimalizačním cyklu hledáme přijatelnou kombinaci většinou jen několika nezávislých parametrů, jež můžeme při návrhu konstrukce ovlivnit. Např. při optimalizaci železobetonového nosníku se dá optimalizovat volba materiálů betonu a výztuže spolu s geometrií průřezu danou obvykle jen několika málo parametry. Požadavky na funkci a architektonický vzhled konstrukce pak dále omezují počet rozměrů optimalizačních úloh ve stavebnictví. Pro účely analýzy s malým počtem vzorků je důležitější, aby algoritmus poskytl kvalitní (nikoli přesné) řešení s využitím nízkého počtu provedených simulací. V programu FNPO je testovaný algoritmus AMS použit v rámci optimalizačního cyklu. Zatím provedené testy se proto soustředí na určení účinnosti navrženého algoritmu při optimalizaci problémů o maximálně 10 dimenzích s malým počtem provedených simulací.

Pro účely testování optimalizačního algoritmu AMS byl využit program FNPO popsaný v kapitole 2. Schopnost algoritmu nalézt globální minimum byla testována na funkcích, které běžně slouží k optimalizačním testům. Optimalizační úlohy pro testování byly definovány jako neohraničená minimalizace funkční hodnoty dle vztahu (59).

3.1 TESTOVACÍ FUNKCE

Kvalita optimalizačních postupů (známých i nově navrhovaných) bývá v literatuře často demonstrována testováním jejich účinnosti na standardních dobře popsaných spojitých funkcích. Běžně užívané funkce pro optimalizační testy lze rozdělit do několika tříd:

a) unimodální, konvexní, mnohodimenzionální

b) multimodální, dvojdimenzionální s malým počtem lokálních extrémů

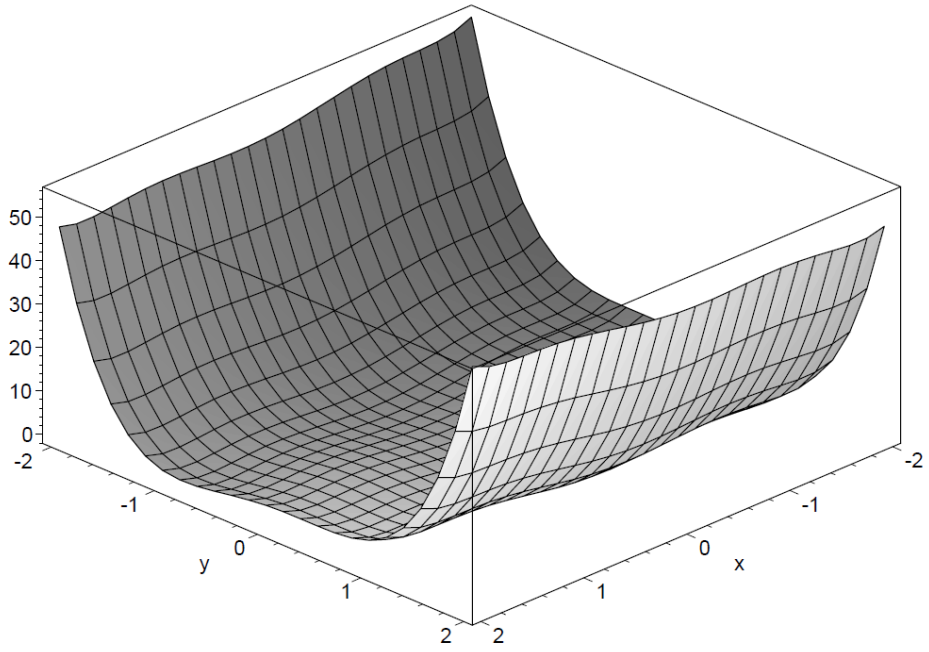
c) multimodální, dvojdimenzionální s velkým počtem lokálních extrémů

d) multimodální, mnohodimenzionální s velkým počtem lokální extrémů

Třída a) zahrnuje širokou skupinu funkcí, od těch, jež jsou velice jednoduché, až po velice náročné funkce způsobující pomalou konvergenci směrem k jedinému funkčnímu extrému. Třída b) tvoří mezistupeň mezi třídami a) a c) - d) a je využívaná k testům schopnosti optimalizačních algoritmů najít globální extrém mezi několika lokálními extrémy. Problémy třídy c) a d) jsou považovány za velice náročné optimalizační testy a slouží k testování pokročilých optimalizačních metod (Genetické algoritmy, Simulované žíhání).

Z časových důvodů bylo zatím provedeno pouze několik testů algoritmu AMS. Následující výčet testovacích funkcí proto reprezentuje pouze ty funkce, na nichž byl algoritmus AMS doposud testován. Další příklady běžných testovacích funkcí užívaných k optimalizačním testům je možné nalézt např. v [52].

SIX-HUMP CAMEL BACK FUNCTION



Obr. 33 Six-hump camel back function – převzato z [52]

Six-hump camel back function je testovací funkce třídy b). Uvnitř ohraničené oblasti se nachází 6 extrémů z toho 2 globální. Funkce je pouze dvojdimenzionální a je popsána následující definicí:

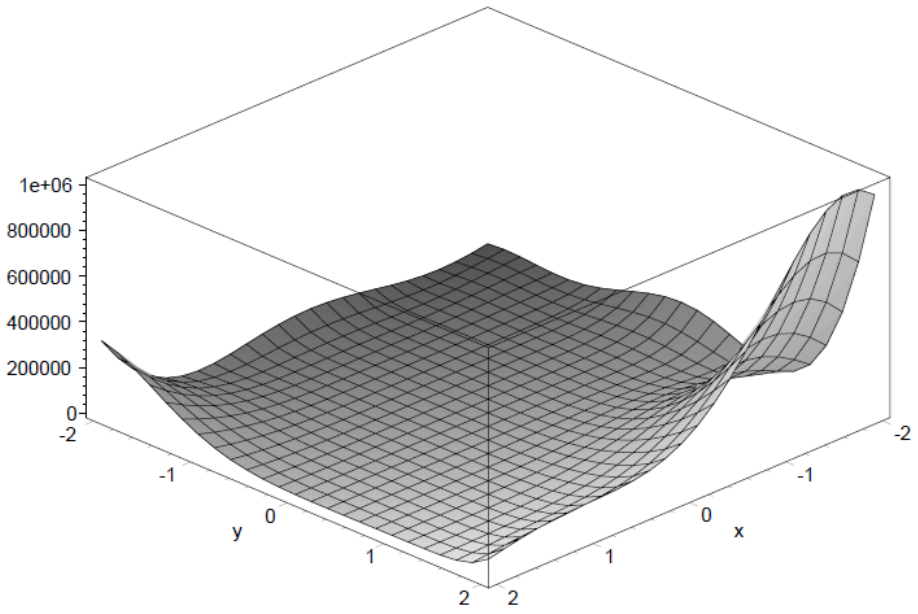
$$f(x_1, x_2) = \left(4 - 2{,}1x_1^2 + \frac{x_1^4}{3}\right)x_1^2 + x_1x_2 + (-4 + 4x_2^2)x_2^2 \qquad (67)$$

Řádkový zápis:

(4-2.1*x1^2+x1^4/3)*x1^2+x1*x2+(-4+4*x2^2)*x2^2

Návrhový prostor je obvykle omezen na obdélník $-3 \leq x_1 \leq 3, -2 \leq x_2 \leq 2$. Globální minima jsou rovna $f(x_1, x_2) = -1{,}03163$, umístěna v bodech$(x_1, x_2) = (-0{,}0898; 0{,}7126)$ a $(x_1, x_2) = (0{,}0898; -0{,}7126)$.

GOLDSTEIN-PRICE'S FUNCTION



Obr. 34 Goldstein-Price's function – převzato z [52]

Je dvojdimenzionální unimodální testovací funkce s minimem umístěným v rozsáhlé ploché oblasti definovaná dle vztahu (68):

$$f(x_1, x_2) = [1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)] \times [30 + (2x_1 - 3x_2)^2(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)] \quad (68)$$

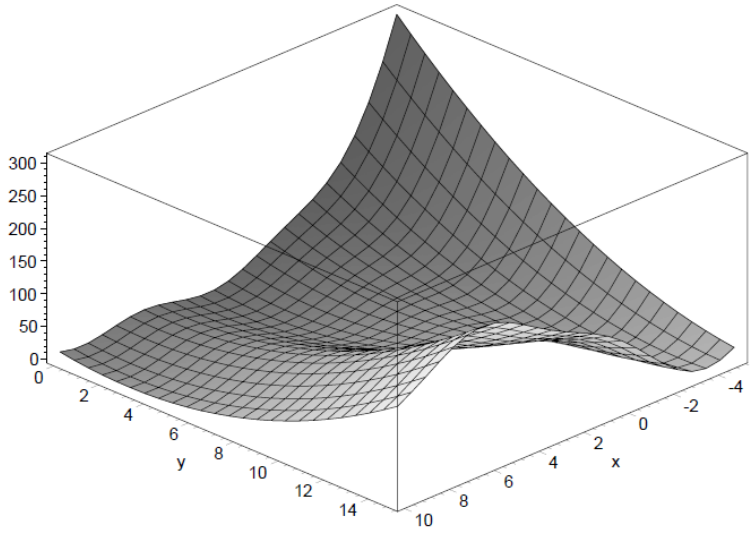
Řádkový zápis:

(1+(x1+x2+1)^2*(19-14*x1+3*x1^2-14*x2+6*x1*x2+3*x2^2))*(30+(2*x1-3*x2)^2*(18-32*x1+12*x1^2+48*x2-36*x1*x2+27*x2^2))

Testovací oblast je omezena na čtverec $-2 \leq x_1 \leq 2, -2 \leq x_2 \leq 2$. Globální minimum je rovno $f(x_1, x_2) = 3$ a je lokalizováno v bodě $(x_1, x_2) = (0, -1)$.

BRANINS'S FUNCTION

Je multimodální testovací funkcí třídy b). Má tři globální extrémy a je definována vztahem (69).



Obr. 35 Branins's function – převzato z [52]

$$f(x_1, x_2) = \left(x_2 - \frac{5{,}1}{4\pi^2}x_1^2 + \frac{5}{\pi} - 6\right) + 10\left(1 - \frac{1}{8\pi}\right)\cos(x_1) + 10 \tag{69}$$

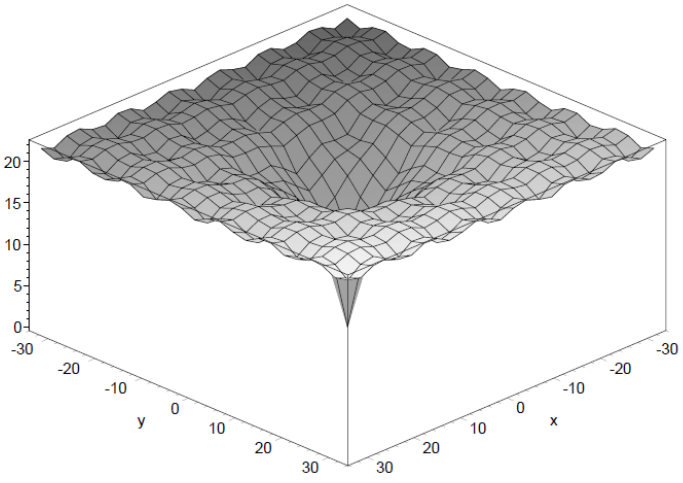
Řádkový zápis:

(x2-(5.1/(4*pi^2))*x1^2+5*x1/pi-6)^2+10*(1-1/(8*pi))*cos(x1)+10

Návrhový prostor je omezen hodnotami $-5 \leq x_1 \leq 10, 0 \leq x_2 \leq 15$. Tři globální minima mají funkční hodnotu $f(x_1, x_2) = 0{,}397887$ a jsou umístěna v bodech:

$$(x_1, x_2) = (-\pi, 12{,}275), (x_1, x_2) = (\pi, 2{,}275), (x_1, x_2) = (9{,}42478, 2{,}475)$$

ACKLEY'S FUNCTION



Obr. 36 Ackley's function – převzato z [52]

Ackley's function je často užívanou multimodální testovací funkcí třídy d). Má následující definici:

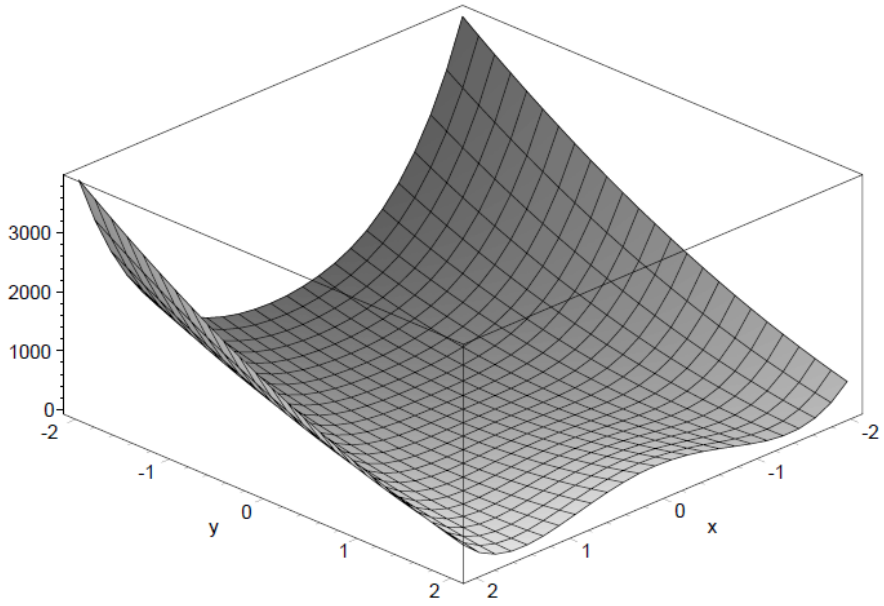
$$f(\boldsymbol{X}) = -20exp\left(-0{,}2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - exp\left(\frac{1}{n}\sum_{i=1}^{n} cos(2\pi x_i)\right) + 20 + e \tag{70}$$

kde počet dimenzí roste od *i=1* do *n*. Návrhový prostor je obvykle omezen na hyperkrychli $-32 \leq x_i \leq 32$. Globální minimum má funkční hodnotu $f(\boldsymbol{X}) = 0$ a je situováno v bodě $x_i = 0$ pro *i=1, ..., n*.

Řádkový zápis (ve 2D):

-20*exp(-0.2*((1/2)*(x1^2+x2^2))^(1/2))-exp((1/2)*(cos(2*pi*x1)+cos(2*pi*x2)))+20+e

ROSENBROCK'S VALLEY



Obr. 37 Rosenbrock's valley – převzato z [52]

Je klasický optimalizační problém také známý jako banana function nebo druhá De Jongova funkce. Globální minimum leží uvnitř dlouhého plochého údolí parabolického tvaru. Poloha optima způsobuje pomalou konvergenci většiny optimalizačních algoritmů. Rosenbrock's valley patří mezi obtížnější často užívané optimalizační testy.

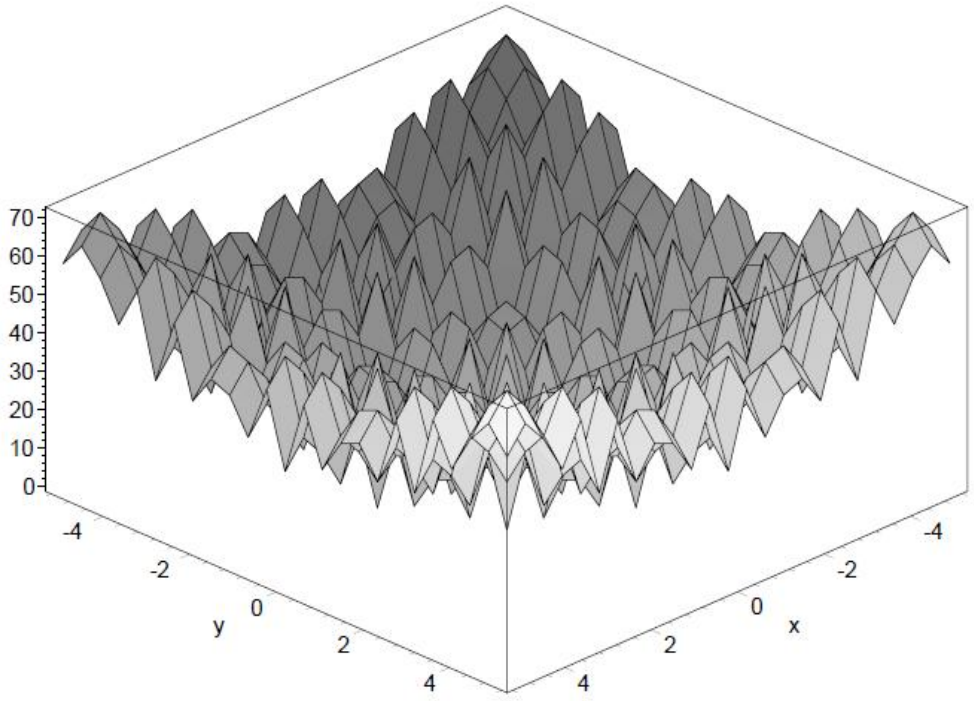
$$f(\boldsymbol{X}) = \sum_{i=1}^{n-1} [100(x_{i+1} - x_i^2) + (1 - x_i)^2] \tag{71}$$

Řádkový zápis (ve 2D):

100*(x2-x1^2)^2+(1-x1)^2

Testovací oblast je obvykle omezena na hyperkrychli $-2{,}048 \leq x_i \leq 2{,}048$, pro *i=1, ..., n.* Globální minimum má funkční hodnotu $f(\boldsymbol{X}) = 0$ a je lokalizováno na souřadnicích $x_i=1$, *i=1, ..., n.*

RASTRIGIN'S FUNCTION



Obr. 38 Rastrigin's function – převzato z [52]

Jedná se o vysoce multimodální testovací funkci třídy d). Jednotlivá lokální minima jsou pravidelně rozprostřena po celém návrhovém prostoru. Funkce má následující definici:

$$f(\boldsymbol{X}) = 10n + \sum_{i=1}^{n} [x_i^2 - 10\cos(2\pi x_i)] \tag{72}$$

Řádkový zápis (ve 2D):

10*2+(x1^2-10*cos(2*pi*x1))+(x2^2-10*cos(2*pi*x2))

Návrhový prostor je zpravidla omezen na hyperkrychli $-5{,}12 \leq x_i \leq 5{,}12$ pro *i=1, ..., n.* Globální minimum je rovno $f(\boldsymbol{X}) = 0$ a je umístěno v bodě $x_i=0$, *i=1, ..., n.*

3.2 VÝSLEDKY TESTOVÁNÍ AMS

Během testování AMS bylo při nastavení vstupních parametrů algoritmu využíváno vztahu (29). Následující kapitola zahrnuje soubor výsledků doposud provedených testů na výše zmíněných testovacích funkcích. U všech provedených testů byla pro simulaci v rámci jednotlivých úrovní využita metoda LHS median. Pro všechny náhodné proměnné bylo definováno rovnoměrné rozdělení (z důvodu rovnoměrného pokrytí návrhového prostoru).

Tabulka 2 je zdrojem dat zobrazených v grafu 6 (str. 38). Ukazuje srovnání algoritmu AMS nastaveného nesprávně - 5 úrovní po 30 simulacích s hodnotou $q = 0{,}5$ a AMS se správným nastavením pro Six-hump camel back function – 12 úrovní po 10 simulacích a $q = 0{,}5$.

Tab. 2 Nesprávné vs. správné nastavení AMS - Six-hump camellback function

Nesprávné vs. správné nastavení AMS - Six-hump camellback function							
Nesprávné nastavení AMS - 150 sim				Správné nastavení AMS - 120 sim			
pokus	min	nalezené min	odchylka	pokus	min	nalezené min	odchylka
1	-1.03163	-1.02756	0.00407	1	-1.03163	-1.03163	0
2	-1.03163	-1.02798	0.00365	2	-1.03163	-1.03163	0
3	-1.03163	-1.03095	0.00068	3	-1.03163	-1.03163	0
4	-1.03163	-1.02685	0.00478	4	-1.03163	-1.03163	0
5	-1.03163	-1.03157	6E-05	5	-1.03163	-1.03163	0
6	-1.03163	-1.03058	0.00105	6	-1.03163	-1.03163	0
7	-1.03163	-1.02235	0.00928	7	-1.03163	-1.03163	0
8	-1.03163	-1.02266	0.00897	8	-1.03163	-1.03163	0
9	-1.03163	-1.03058	0.00105	9	-1.03163	-1.03163	0
10	-1.03163	-1.02413	0.0075	10	-1.03163	-1.03163	0
11	-1.03163	-1.03072	0.00091	11	-1.03163	-1.03163	0
12	-1.03163	-1.02647	0.00516	12	-1.03163	-1.03163	0
13	-1.03163	-1.02644	0.00519	13	-1.03163	-1.03163	0
14	-1.03163	-1.03083	0.0008	14	-1.03163	-1.03163	0
15	-1.03163	-1.02114	0.01049	15	-1.03163	-1.03163	0
16	-1.03163	-1.02857	0.00306	16	-1.03163	-1.03163	0
17	-1.03163	-1.03031	0.00132	17	-1.03163	-1.03163	0
18	-1.03163	-1.03134	0.00029	18	-1.03163	-1.03163	0
19	-1.03163	-1.02807	0.00356	19	-1.03163	-1.03163	0
20	-1.03163	-1.02891	0.00272	20	-1.03163	-1.03163	0
střední hodnota odchylky:				střední hodnota odchylky:			
0.00331				0			
směrodatná odchylka:				směrodatná odchylka:			
0.003133566				0			

Přesnost algoritmu při správném nastavení je dána malou velikostí prohledávaného prostoru v posledním cyklu. Rozbor konvergenční rychlosti v případě Six-hump camel back function ukázal, že optimální počet cyklů pro q = 0,5 bude 12-14. Během testování se však ukázalo, že k dosažení požadované přesnosti dojde již ve 12. cyklu. Z tohoto důvodu byl počet simulací potřebný k dosažení výsledků zobrazených v pravé části tabulky 2 snížen na 120.

Tabulka 3 reprezentuje data zobrazená v grafu 7 (str. 38). Představuje tedy srovnání správného a nesprávného nastavení AMS u Goldstein-Price's function. Nesprávné nastavení odpovídalo 5 úrovním po 30 simulacích s hodnotou $q = 0{,}5$. Správné nastavení AMS pro danou funkci bylo definováno jako: 14 úrovní po 10 simulacích a $q = 0{,}5$.

Tab. 3 Nesprávné vs. správné nastavení AMS - Goldstein-Price's function

Nesprávné vs. Správné nastavení AMS - Goldstein-Price's function							
Nesprávné nastavení AMS 150 sim				Správné nastavení AMS - 140 sim			
pokus	min	nalezené min	odchylka	pokus	min	nalezené min	odchylka
1	3	3.05929	0.05929	1	3	3	0
2	3	3.0081	0.0081	2	3	3	0
3	3	3.0328	0.0328	3	3	3	0
4	3	3.00818	0.00818	4	3	3	0
5	3	3.06016	0.06016	5	3	3	0
6	3	3.05726	0.05726	6	3	3	0
7	3	3.09925	0.09925	7	3	3	0
8	3	3.11844	0.11844	8	3	3	0
9	3	3.08137	0.08137	9	3	3	0
10	3	3.01582	0.01582	10	3	3	0
11	3	3.03282	0.03282	11	3	3	0
12	3	3.05724	0.05724	12	3	3	0
13	3	3.08136	0.08136	13	3	3	0
14	3	3.3272	0.3272	14	3	3	0
15	3	3.00815	0.00815	15	3	3	0
16	3	3.08533	0.08533	16	3	3	0
17	3	3.50456	0.50456	17	3	3	0
18	3	3.17969	0.17969	18	3	3	0
19	3	3.03226	0.03226	19	3	3	0
20	3	3.24284	0.24284	20	3	3	0
střední hodnota odchylky:				střední hodnota odchylky:			
0.059725				0			
směrodatná odchylka:				směrodatná odchylka:			
0.121611059				0			

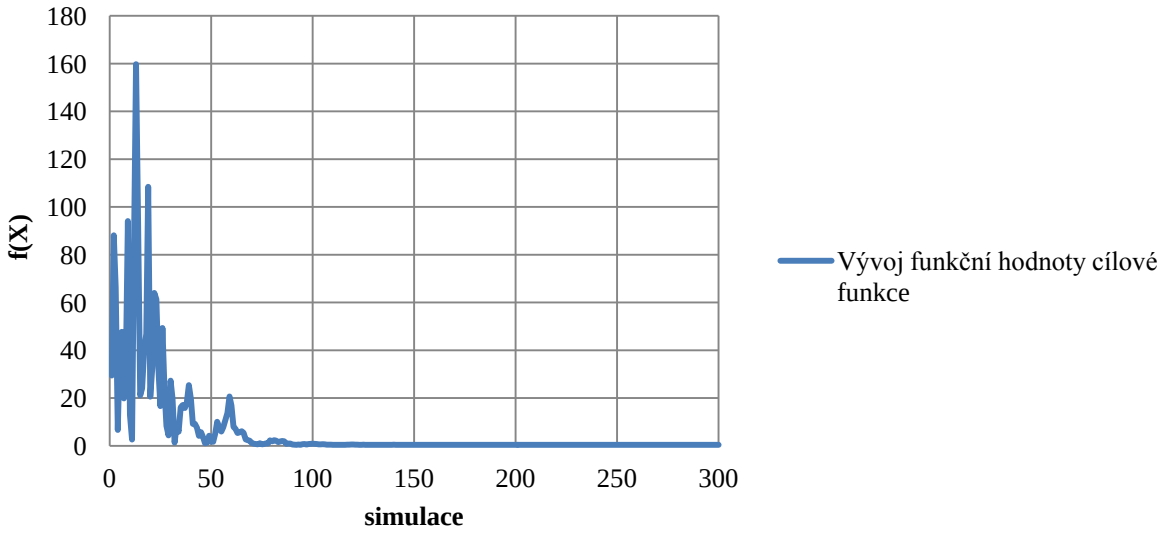
Byly také provedeny testy s nižším počtem simulací (viz graf 9). Provedené testy ukázaly, že algoritmus AMS je u některých testovaných funkcí schopen lokalizovat minimum i s řádově desítkami simulací. Např. u Six-hump camel back function byla při užití pouhých 60 simulací střední hodnota odchylky řešení po provedení 20 optimalizačních testů 0,00003. Při snižování počtu simulací však dochází častěji k úplnému selhání vyhledávání. Rozptyl odchylky řešení se proto se snižujícím se počtem simulací rychle zvyšuje. Při ideálním nastavení algoritmu AMS prezentovaném v tabulkách 2 a 3 pro obě testované funkce také dochází k občasnému selhání vyhledávání. Četnost těchto selhání je však nízká a během 20 testovacích pokusů zaznamenaných v tabulkách 2 a 3 k nim nedošlo.

Tabulka 4 reprezentuje výsledky optimalizace Branin's function algoritmem AMS. Pro zvolenou hodnotu $q = 0,5$ byl dle grafu sestaveného na základě vztahu (29) stanoven optimální počet cyklů $i_{max} = 15$. Celkový počet simulací byl stanoven nejprve na 150, tedy 10 simulací v každém z cyklů. V dalších 20 pokusech byl počet simulací v každém z cyklů zdvojnásoben.

Tab. 4 Optimalizace Branin's function algoritmem AMS

Branin's function - $_{a1}$=15 - 150 sim.				Branin's function - 300 sim.			
pokus	min	nalezené min	odchylka	pokus	min	nalezené min	odchylka
1	0.397887	0.397888	1E-06	1	0.397887	0.397887	0
2	0.397887	0.397887	0	2	0.397887	0.397887	0
3	0.397887	0.808887	0.411	3	0.397887	0.397887	0
4	0.397887	0.397887	0	4	0.397887	0.397887	0
5	0.397887	0.397887	0	5	0.397887	0.397887	0
6	0.397887	0.400300	0.002413	6	0.397887	0.397887	0
7	0.397887	0.399224	0.001337	7	0.397887	0.397887	0
8	0.397887	0.422662	0.024775	8	0.397887	0.397887	0
9	0.397887	0.399824	0.001937	9	0.397887	0.397887	0
10	0.397887	0.397887	0	10	0.397887	0.398381	0.000494
11	0.397887	0.398221	0.000334	11	0.397887	0.398365	0.000478
12	0.397887	0.426993	0.029106	12	0.397887	0.397887	0
13	0.397887	0.404342	0.006455	13	0.397887	0.397887	0
14	0.397887	0.849032	0.451145	14	0.397887	0.397893	6E-06
15	0.397887	0.653165	0.255278	15	0.397887	0.397887	0
16	0.397887	0.397928	4.1E-05	16	0.397887	0.397887	0
17	0.397887	0.403035	0.005148	17	0.397887	0.397887	0
18	0.397887	0.402637	0.00475	18	0.397887	0.397925	3.8E-05
19	0.397887	0.397887	0	19	0.397887	0.397887	0
20	0.397887	0.398838	0.000951	20	0.397887	0.397887	0
střední hodnota odchylky:				střední hodnota odchylky:			
0.001637				0			
směrodatná odchylka:				směrodatná odchylka:			
0.135613753				0.000145325			

Graf 11 zobrazuje postupnou konvergenci algoritmu AMS k hledanému řešení při jednom z případů optimalizace Branin's function s celkovým počtem 300 simulací.



Graf 12 Průběh optimalizačního procesu (Branin's function – 300 simulací)

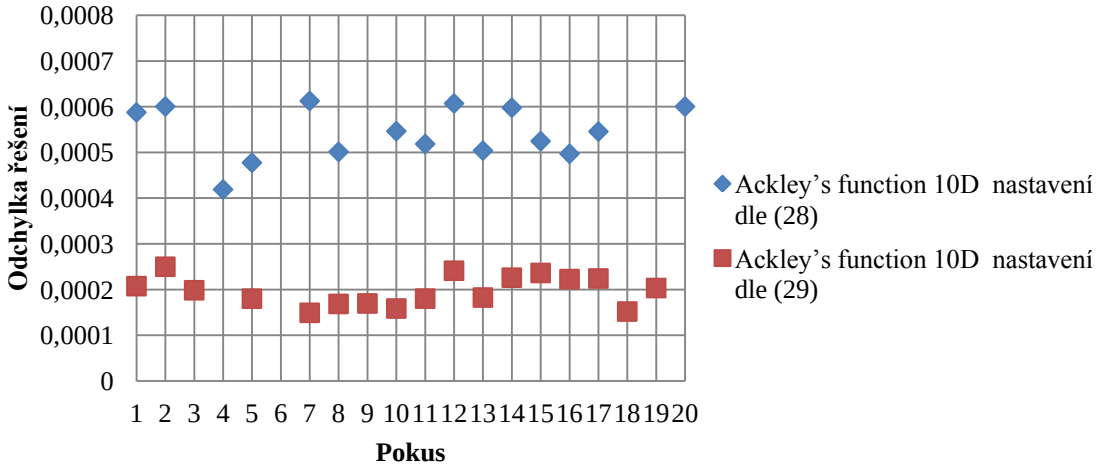
Z grafu 11 je zřejmé, že ke "hrubé" lokalizaci optima zkoumané funkce došlo už po několika prvních cyklech. Provedené testy naznačují, že algoritmus AMS konverguje zpočátku velice rychle. Problém nastává při finálním "zpřesnění" řešení v posledních několika cyklech. Na přesnou identifikaci optima v rámci malého intervalu hodnot tak připadla více než polovina všech provedených simulací.

Na základě dosavadních testů algoritmu AMS na problémech vyšších dimenzí došlo k heuristickému zpřesnění vztahu pro maximální teoretickou rychlost konvergence. Vztah (29) tedy částečně zohledňuje vliv počtu dimenzí na účinné nastavení algoritmu AMS. Při nastavení algoritmu dle vztahu (29) došlo k zpřesnění výsledků optimalizačního procesu u prozatím testovaných problémů vyšších dimenzí přibližně o přibližně 60 – 70% oproti původnímu nastavení dle vztahu (28). Algoritmus AMS tedy poskytuje relativně stabilní řešení i u vícedimenzionálních problémů, jeho přesnost je však nižší než přesnost pokročilých verzí evolučních algoritmů [35]. Tento fakt může být zapříčiněn nedostatečnou korektností heuristicky upraveného vztahu (29). Možné zpřesnění vztahu (29) a zlepšení nastavení algoritmu AMS bude předmětem dalšího výzkumu.

Tabulka 5 demonstruje srovnání testů algoritmu AMS při optimalizaci Ackley's function v desetirozměrném návrhovém prostoru. Levá strana tabulky obsahuje výsledky optimalizace získané při nastavení AMS dle vztahu (28) – 23 cyklů po 1174 simulacích, celkem tedy 27002 simulací při $q=0,6$. Pravá strana tabulky obsahuje výsledky optimalizace při nastavení AMS dle vztahu (29) – 27 cyklů po 1000 simulacích při $q = 0,6$. V grafu 13 zobrazujícím výsledná data tabulky 5 byly z důvodu přehlednosti zanedbány ty realizace optimalizačních pokusů, které se odchýlily od přesného řešení o více než 0,0008.

Tab. 5 Srovnání nastavení algoritmu AMS dle vztahů (28) a (29) (Ackley's function 10D)

Ackley's function 10D nastavení dle (28)				Ackley's function 10D nastavení dle (29)			
pokus	min	nalezené min	odchylka	pokus	min	nalezené min	odchylka
1	0	0.000587	0.00058745	1	0	0.000207	0.00020732
2	0	0.0006	0.00060042	2	0	0.00025	0.00024975
3	0	1.64622	1.64622	3	0	0.000198	0.00019825
4	0	0.000419	0.00041884	4	0	1.64622	1.64622
5	0	0.000477	0.00047737	5	0	0.00018	0.00017984
6	0	0.872429	0.872429	6	0	0.07227	0.0722702
7	0	0.000612	0.00061236	7	0	0.000149	0.00014896
8	0	0.000501	0.00050111	8	0	0.000168	0.00016834
9	0	1.15515	1.15515	9	0	0.000169	0.00016941
10	0	0.000546	0.00054634	10	0	0.000158	0.00015833
11	0	0.000518	0.00051826	11	0	0.00018	0.00018
12	0	0.000607	0.00060697	12	0	0.000241	0.00024103
13	0	0.000504	0.0005038	13	0	0.000182	0.00018236
14	0	0.000598	0.0005977	14	0	0.000226	0.0002258
15	0	0.000524	0.00052445	15	0	0.000236	0.00023608
16	0	0.000497	0.00049693	16	0	0.000223	0.0002227
17	0	0.000546	0.0005456	17	0	0.000224	0.00022407
18	0	1.64622	1.64622	18	0	0.000152	0.00015174
19	0	1.64622	1.64622	19	0	0.000203	0.00020316
20	0	0.0006	0.00060021	20	0	1.15516	1.15516
střední hodnota odchylky:				střední hodnota odchylky:			
0.000592573				0.000205239			
směrodatná odchylka:				směrodatná odchylka:			
0.624240844				0.426368555			



Graf 13 Srovnání nastavení algoritmu AMS dle vztahů (28) a (29) (Ackley's function 10D)

Rastrigin's function představuje náročný multimodální optimalizační test. Tabulka 6 zobrazuje výsledky testování algoritmu AMS na této funkci. Test slouží k demonstraci schopnosti algoritmu nalézt globální minimum v náročném prostředí s velkým počtem ostrých lokálních extrémů. Byly provedeny dvě série testů po 20 optimalizačních pokusech při dvou rozdílných nastaveních algoritmu AMS. První nastavení odpovídalo požadavkům pro SSA a nezohledňovalo tvar funkce, který není v obecném případě optimalizace předem znám. Bylo užito celkem 300 simulací v 15 cyklech po 20 simulacích s hodnotou $q = 0,5$. Druhý způsob nastavení měl zaručit nalezení optima s velkou pravděpodobností. Cílení proto probíhalo pomalu s koeficientem $q = 0,8$, pro nějž byl stanoven optimální počet cyklů $i_{max}=42$. V každém z cyklů bylo provedeno 100 simulací, celkem tedy 4200 simulací.

Tab. 6 Test Rastrigin's function

Rastrigin's function 2D AMS a_1=10.24 320 sim.				Rastrigin's function 2D AMS a_1=10.24 4200 sim.			
pokus	min	nalezené min	odchylka	pokus	min	nalezené min	odchylka
1	0	4.18E-07	4.176E-07	1	0	1.99E-01	0.199013
2	0	1.99E+00	1.98992	2	0	7.75E-08	7.75E-08
3	0	4.97E+00	4.97479	3	0	1.28E+00	1.27886
4	0	3.98E+00	3.97983	4	0	7.71E-01	0.770604
5	0	1.67E-06	1.671E-06	5	0	6.28E-09	6.28E-09
6	0	1.98992	1.98992	6	0	0.080011	0.080011
7	0	0.99496	0.99496	7	0	0.171402	0.171402
8	0	0.994959	0.994959	8	0	0.655857	0.655857
9	0	9.95E-01	0.994961	9	0	2.44E-07	2.44E-07
10	0	0.994962	0.994962	10	0	2.64E-08	2.64E-08
11	0	9.95E-01	0.99496	11	0	6.94E-08	6.94E-08
12	0	1.18E-06	1.175E-06	12	0	3.26E-07	3.26E-07
13	0	5.08E-07	5.08E-07	13	0	2.18E-07	2.18E-07
14	0	9.95E-01	0.99496	14	0	9.95E-01	0.994959
15	0	1.98992	1.98992	15	0	3.46E-08	3.46E-08
16	0	9.09E-07	9.092E-07	16	0	1.08E-07	1.08E-07
17	0	9.95E-01	0.994959	17	0	7.29E-01	0.729487
18	0	2.75E-06	2.749E-06	18	0	1.22E-01	0.121917
19	0	0.99496	0.99496	19	0	5.86E-08	5.86E-08
20	0	0.994962	0.994962	20	0	1.96E-07	1.96E-07
střední hodnota odchylky:				střední hodnota odchylky:			
0.99496				2.85285E-07			
směrodatná odchylka:				směrodatná odchylka:			
1.281913356				0.388635497			

Rastrigin's function má jedno globální minimum, jež je obklopeno několika blízkými lokálními minimy s funkční hodnotou 0,99459 tato blízká minima mají jednu ze svých souřadnic shodnou s minimem globálním. V případě nastavení algoritmu AMS

odpovídajícímu výsledkům v levé části tabulky 6 došlo ke správnému nalezení minima v 6 z 20 testů. V 9 testech pak algoritmus skončil v lokálních minimech v bezprostředním okolí minima globálního. U zbylých 5 testů pak vyhledávání skončilo v některém ze vzdálenějších lokálních minim. Při užití nastavení odpovídajícího pravé části tabulky 6 došlo k nalezení minima s přesností na 7 desetinných míst v 11 z 20 testů. Zohledníme – li tvar funkce (Obr. 38) pak můžeme říci, že v případech kdy nalezená funkční hodnota klesla pod hodnotu 0,99459 odpovídající lokáním extrémům v bezprostřední blízkosti globálního minima, lokalizoval algoritmus "údolí" funkce, na jehož dně se globální minimum nachází. Takových případů je ve výsledcích z pravé části uvedené tabulky 18 z 20 testů.

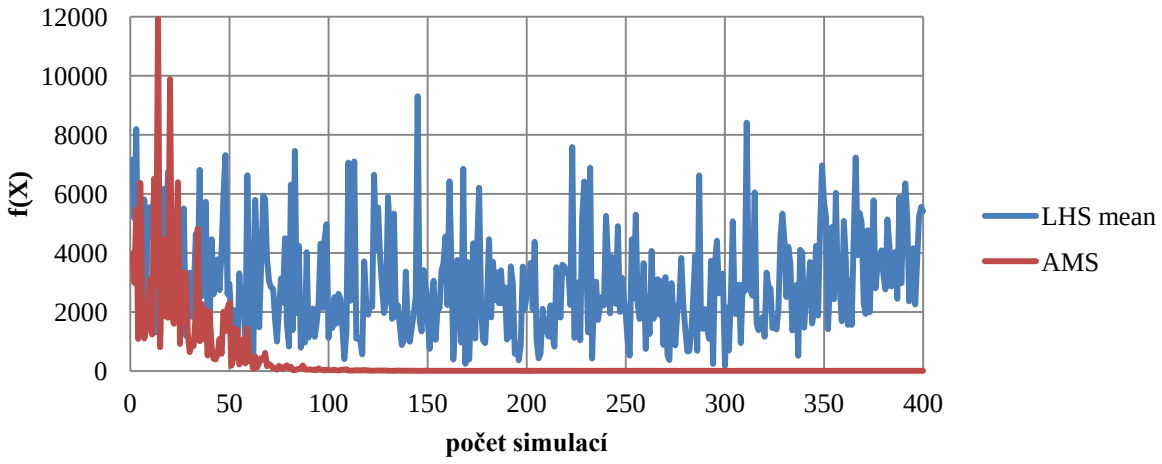
Předběžné testy na problémech vyšších dimenzí (10 a více) naznačují, že pokročilé verze evolučních algoritmů budou u těchto úloh dosahovat lepší přesnosti řešení než algoritmus AMS se srovnatelným počtem provedených simulací. Při požadavku na přesnou lokalizaci extrému je počet simulací potřebný k řešení mnohodimenzionálních optimalizačních problémů vysoký u všech známých optimalizačních technik. Užití desítek tisíc simulací obvykle potřebných pro optimalizaci úloho řadu 10 a více neodpovídá požadavku na analýzu s malým počtem vzorků. Zda algoritmus AMS poskytne lepší řešení mnohodimenzionálních problémů pří užití řádově stovek až tisíců simulací než pokročilé evoluční algoritmy [35] se stejnými podmínkami bude předmětem dalšího zkoumání.

Při práci s evolučními algoritmy je nejprve třeba vytvořit počáteční generaci vzorků v rámci návrhového prostoru. K těmto účelům se dnes využívá například metod typu LHS. Nabízí se tedy možnost vytvořit počáteční generaci metodou AMS. Takto vytvořená počáteční generace vzorků by pak mohla disponovat lepšími vlastnostmi než vzorky vygenerované prostou simulací v rámci návrhového prostoru. Pro aplikaci pokročilých metod evolučních strategií by tak byla připravena lepší výchozí pozice. Jinou možností by bylo použití nejlepšího vzorku získaného algoritmem AMS s užitím malého počtu simulací jako výchozího bodu pro start metody simulovaného žíhání.

Rosenbrock's valley je náročným optimalizačním problémem. K jeho přesnému řešení ve více než 6 dimenzích potřebuje většina pokročilých optimalizačních algoritmů řádově statisíce simulací. Předpokládejme nyní vytvoření první generace 400 vzorků v rámci daného návrhového prostoru. Pro účel srovnání generace vytvořené prostou simulací metodou LHS a generace vytvořené metodou AMS se stejným celkovým množstvím simulací byl proveden následující test. Ve 20 nezávislých pokusech bylo nejprve generováno 400 simulací metodou LHS mean v rámci sedmirozměrného návrhového prostoru funkce Rosenbrock's valley. Poté bylo provedeno 20 optimalizačních pokusů s využitím algoritmu AMS. Funkce Rosenbrock's valley je přes svou náročnost funkcí unimodální. Nehrozí proto zastavení algoritmu v lokálním extrému. Tato skutečnost by však v obecném případě optimalizace nebyla známa. Z tohoto důvodu bylo pro algoritmus AMS voleno volnější tempo cílení dané volbou hodnoty $q=0,8$. Pro tuto hodnotu byl dle vztahu (29) určen optimální počet cyklů $i_{max}=40$. V každém z cyklů proběhlo 10 simulací. Výsledné srovnání obou postupů na základě nejlepších získaných hodnot je k dispozici v tabulce 7. Srovnání dvou získaných populací odpovídajících prvnímu řádku tabulky 7 (pokus 1) je zobrazeno v grafu 14.

Tab. 7 Srovnání výsledků simulace metodou LHS a AMS (Rosenbrock's valley – 400 sim)

Rosenbrock's valley 7D LHS - 400 sim				Rosenbrock's valley 7D AMS - 400 sim			
pokus	min	nalezené min	odchylka	pokus	min	nalezené min	odchylka
1	0	98.3528	98.3528	1	0	5.38472	5.38472
2	0	195.514	195.514	2	0	5.10997	5.10997
3	0	96.1837	96.1837	3	0	6.3564	6.3564
4	0	239.971	239.971	4	0	5.32597	5.32597
5	0	254.209	254.209	5	0	6.54453	6.54453
6	0	141.479	141.479	6	0	3.02715	3.02715
7	0	440.678	440.678	7	0	79.3355	79.3355
8	0	233.144	233.144	8	0	3.32834	3.32834
9	0	124.165	124.165	9	0	5.35654	5.35654
10	0	180.194	180.194	10	0	6.18034	6.18034
11	0	140.819	140.819	11	0	5.07007	5.07007
12	0	283.26	283.26	12	0	6.40082	6.40082
13	0	168.802	168.802	13	0	6.10629	6.10629
14	0	171.969	171.969	14	0	0.096355	0.096355
15	0	124.022	124.022	15	0	5.19191	5.19191
16	0	194.276	194.276	16	0	12.4759	12.4759
17	0	257.383	257.383	17	0	65.0398	65.0398
18	0	155.206	155.206	18	0	6.2234	6.2234
19	0	100.505	100.505	19	0	6.90213	6.90213
20	0	177.292	177.292	20	0	4.47314	4.47314
střední hodnota odchylky:				střední hodnota odchylky:			
174.6305				5.745505			
směrodatná odchylka:				směrodatná odchylka:			
79.21334256				20.243537			



Graf 14 srovnání 400 vzorků generovaných LHS mean a AMS

SROVNÁNÍ ALGORITMU AMS A JINÝCH OPTIMALIZAČNÍCH METOD

Součástí budoucích plánovaných testů by mělo být i srovnání účinnosti metody AMS a jiných pokročilých optimalizačních technik zahrnujících např. evoluční algoritmy či metodu Simulovaného žíhání. První testy srovnávající účinnost algoritmu AMS a metody Simulovaného žíhání již na výše uvedených optimalizačních problémech provedeny byly. V literatuře jsou sice dostupné příklady výsledků optimalizace metodou Simulovaného žíhání pro výše uvedené testovací funkce, těchto výsledků však bylo u drtivé většiny dostupných příkladů dosaženo až po náročném heuristickém hledání optimálního nastavení algoritmu pro daný výchozí bod. Z důvodu objektivního posouzení účinnosti metody Simulovaného žíhání byla provedena nová (zatím velice omezená) série testů. K testování byl využit algoritmus zapsaný v programu Matlab převzatý z [59]. Tabulka 8 obsahuje výsledky optimalizace metodou simulovaného žíhaní pro Six-hump camel back function.

Tab. 8 Test simulovaného žíhání Six-hump camel back function

Six-hump camel back function - Simulované žíhání				
pokus	min	nalezené min	odchylka	počet vyčíslení f(**X**)
1	-1.03163	-1.03163	0	7613
2	-1.03163	2.10425	3.13588	7604
3	-1.03163	2.10425	3.13588	6195
4	-1.03163	-0.21546	0.816167	6716
5	-1.03163	-1.03163	0	6633
6	-1.03163	-0.21546	0.816167	6646
7	-1.03163	-1.03163	0	6309
8	-1.03163	-1.03163	0	6771
9	-1.03163	-0.21546	0.816167	6423
10	-1.03163	2.10425	3.13588	6399
11	-1.03163	-1.03163	0	6094
12	-1.03163	-0.21546	0.816166	6878
13	-1.03163	-0.21546	0.816166	8134
14	-1.03163	-0.21546	0.816166	7342
15	-1.03163	-0.21546	0.816166	6828
16	-1.03163	2.10425	3.13588	6163
17	-1.03163	-0.21546	0.816166	7490
18	-1.03163	2.10425	3.13588	6100
19	-1.03163	-0.9925	0.03913	5179
20	-1.03163	-0.9906	0.04103	5855
střední hodnota odchylky:				průměr:
0.816166				6668.6
směrodatná odchylka:				
1.217809895				

Během testování bylo využíváno doporučené nastavení pro optimalizaci obecné funkce neznámého tvaru dle [59]:

- Funkce ochlazování: $T_n=0.8T_o$
- Funkce mutace: přičtení malého náhodného čísla k jedné ze souřadnic náhodného vektoru
- Počáteční teplota: T=1
- Max. počet po sobě jdoucích zamítnutí: 1000
- Max. počet vzorků přijatých při stejné T: 20
- Max. počet iterací při stejné T: 300
- Konečná teplota: T = 1e-8

Podrobný popis užitého algoritmu je k dispozici v [59].

Výše uvedené nastavení metody Simulovaného žíhání bylo v [59] použito k úspěšné lokalizaci extrému Six-hump camel back function za definice výchozího bodu (0, 0). Při optimalizaci obecné funkce neznámého tvaru však nemáme informace o kvalitě nalezeného řešení a nemůžeme proto ladit nastavení parametrů metody Simulovaného žíhání vzhledem k užitému počátečnímu bodu. Volba výchozího bodu je v obecném případě náhodná. Během testů byl proto výchozí návrhový vektor v každém z pokusů nově vygenerován simulací Monte Carlo. Z tabulky 8 je zřejmé, že algoritmus AMS byl při optimalizaci Six-hump camel back function mnohem účinnější než metoda Simulovaného žíhání. Lepších výsledků co do průměrné odchylky dosahuje Simulované žíhání s použitím jiné funkce mutace - násobení návrhového vektoru vektorem souřadnic bodu náhodně vygenerovaného v jednotkové hyperkrychli. Takto získaná řešení sice dávají menší průměrnou odchylku, ale ani v jednom z testovaných případů se nepodařilo lokalizovat minimum přesně. Počet provedených simulací pak vždy přesáhl hodnotu 5000.

Cílem dalšího z testů bylo srovnání účinnosti AMS a Simulovaného žíhání při SSA. Při předběžných testech na dalších výše uvedených dvourozměrných funkcích se ukázalo, že metoda AMS poskytuje lepší či stejně kvalitní řešení jako (obecně nastavená) metoda Simulovaného žíhání s mnohem menším počtem provedených simulací. U problémů vyšších dimenzí nemusí být srovnání obou metod takto jednoznačné. Pro test SSA byla proto zvolena funkce Rosenbrock's valley (7D). Nastavení algoritmu Simulovaného žíhání bylo upraveno tak, aby celkový počet provedených simulací nepřesáhl hodnotu 440. Pro účely SSA je také vhodnější volit jako funkci mutace násobení návrhového vektoru vektorem souřadnic bodu náhodně vygenerovaného v jednotkové hyperkrychli. Tato funkce zajistí rychlejší počáteční konvergenci algoritmu. Vzhledem náročnosti optimalizačního testu a nízkému počtu provedených simulací není možné očekávat přesnou lokalizaci minima funkce. Cílem je srovnat funkční hodnotu získanou AMS a metodou Simulovaného žíhání s malým omezeným počtem vzorků. Pro účely srovnání byla k levé části tabulky 9 reprezentující výsledky optimalizace metodou Simulovaného žíhání přidána pravá část tabulky 7 s výsledky optimalizace stejného problému metodou AMS se srovnatelným počtem simulací (400).

Tab. 9 Srovnání AMS a Simulovaného žíhání při SSA (Rosenbrock's valley 7D)

Rosenbrock's valley - Simulované žíhání - SSA					Rosenbrock's function 7D AMS - 400 sim			
pokus	min	nalezené min	odchylka	počet vyčíslení f(**X**)	pokus	min	nalezené min	odchylka
1	0	2.77E+01	27.7118	440	1	0	5.38472	5.38472
2	0	17.6608	17.6608	440	2	0	5.10997	5.10997
3	0	16.3861	16.3861	440	3	0	6.3564	6.3564
4	0	12.4163	12.4163	440	4	0	5.32597	5.32597
5	0	6.5804	6.5804	440	5	0	6.54453	6.54453
6	0	12.0812	12.0812	440	6	0	3.02715	3.02715
7	0	11.2996	11.2996	440	7	0	79.3355	79.3355
8	0	24.8558	24.8558	440	8	0	3.32834	3.32834
9	0	18.2183	18.2183	440	9	0	5.35654	5.35654
10	0	9.1415	9.1415	440	10	0	6.18034	6.18034
11	0	19.2207	19.2207	440	11	0	5.07007	5.07007
12	0	18.8939	18.8939	440	12	0	6.40082	6.40082
13	0	20.7897	20.7897	440	13	0	6.10629	6.10629
14	0	20.7228	20.7228	440	14	0	0.0963553	0.0963553
15	0	12.8813	12.8813	440	15	0	5.19191	5.19191
16	0	6.425	6.425	440	16	0	12.4759	12.4759
17	0	15.4826	15.4826	440	17	0	65.0398	65.0398
18	0	15.0137	15.0137	440	18	0	6.2234	6.2234
19	0	26.7338	26.7338	440	19	0	6.90213	6.90213
20	0	24.9749	24.9749	440	20	0	4.47314	4.47314
střední hodnota odchylky:				průměr:	střední hodnota odchylky:			
17.02345				440	5.745505			
směrodatná odchylka:					směrodatná odchylka:			
6.150569721					20.24353737			

Je vidět, že metoda AMS poskytuje u problému Rosenbrock's valley o 7 dimenzích kvalitnější řešení v rámci SSA než metoda Simulovaného žíhání s výše uvedeným nastavením. Během testování došlo u AMS 2x k výraznější odchylce od přesného řešení. To způsobilo vyšší hodnotu uvedené směrodatné odchylky algoritmu AMS. Při užití funkce mutace aplikované u Simulovaného žíhání v testu SSA zachyceném v tabulce 9 nedojde k výraznému zpřesnění ani při odstranění omezení maximálního počtu simulací.

Porovnejme nyní účinnost metody Simulovaného žíhání v nastavení pro optimalizaci neznámé funkce (s užitím funkce mutace umožňující přesnější lokalizaci extrému) s účinností algoritmu AMS pracujícím s počtem simulací, který je srovnatelný s průměrným počtem simulací užitých při Simulovaném žíhání. Nejprve byl proveden test metody Simulovaného žíhání. Poté byl stanoven průměrný počet užitých simulací a na jeho základě bylo definováno

nastavení algoritmu AMS dle (29) s celkovým počtem simulací 5400. Výsledek testu je zachycen v tabulce 10.

Tab. 10 Srovnání AMS a Simulovaného žíhání (Rosenbrock's valley 7D)

Rosenbrock's valley - Simulované žíhání - SSA					Rosenbrock's function 7D AMS - 5400 sim			
pokus	min	nalezené min	odchylka	počet vyčíslení f(**X**)	pokus	min	nalezene min	odchylka
1	0	3.74E+00	3.73747	4234	1	0	4.04615	4.04615
2	0	3.39984	3.39984	4621	2	0	6.55671	6.55671
3	0	3.57068	3.57068	5828	3	0	2.54271	2.54271
4	0	3.67972	3.67972	5528	4	0	2.88613	2.88613
5	0	3.73616	3.73616	4587	5	0	5.03953	5.03953
6	0	0.212418	0.212418	7824	6	0	2.01539	2.01539
7	0	4.7179	4.7179	4859	7	0	4.10566	4.10566
8	0	4.04374	4.04374	4546	8	0	3.98992	3.98992
9	0	19.4614	19.4614	3577	9	0	4.50701	4.50701
10	0	6.55405	6.55405	8984	10	0	3.69566	3.69566
11	0	3.62194	3.62194	4206	11	0	4.36384	4.36384
12	0	4.01404	4.01404	12795	12	0	1.50305	1.50305
13	0	6.57741	6.57741	4530	13	0	5.53391	5.53391
14	0	3.36412	3.36412	5697	14	0	3.32433	3.32433
15	0	14.9751	14.9751	4156	15	0	6.61494	6.61494
16	0	2.96217	2.96217	4759	16	0	1.95291	1.95291
17	0	1.77875	1.77875	5565	17	0	0.762451	0.762451
18	0	4.59814	4.59814	5322	18	0	0.261518	0.261518
19	0	2.51172	2.51172	5213	19	0	5.36304	5.36304
20	0	5.27497	5.27497	4181	20	0	1.88744	1.88744
střední hodnota odchylky:				průměr:	střední hodnota odchylky:			
3.736815				5550.6	3.84279			
směrodatná odchylka:					směrodatná odchylka:			
4.319179662					1.763154774			

Je zřejmé, že u dané funkce vykazují oba algoritmy srovnatelnou účinnost. Metoda AMS však poskytuje stabilnější řešení. Na rozdíl od metody Simulovaného žíhání není výsledek procesu optimalizace přímo závislý na nahodilé volbě jediného výchozího bodu. Nutno podotknout, že Rosenbrock's valley je náročným optimalizačním problémem, při jehož optimalizaci dochází k velice pomalé konvergenci k přesnému řešení u většiny pokročilých optimalizačních metod [35].

Bylo také provedeno několik testů s různými variantami nastavení metody Simulovaného žíhání u problému Ackley's function o deseti dimenzích řešeného algoritmem AMS stabilně s přesností na 3 desetinná místa s užitím 27000 simulací. Ani v jednom

z provedených pokusů se však metodě Simulovaného žíhání spolehlivě nalézt optimum nepodařilo.

Srovnání metody AMS s pokročilými verzemi genetických algoritmů bude jedním z úkolů budoucího testování. Pro účely testování zatím nebyla k dispozici vhodná automatizace algoritmů popsaných např. v [35]. Nebylo proto možné provést předběžné srovnávací testy podobně jako u metody Simulovaného žíhání. Částečné srovnání metody AMS a pokročilých evolučních algoritmů je tedy možné pouze na základě výsledků testů shodných optimalizačních problémů uváděných v literatuře. Např. pro Six-hump camel back function uvádí [35], že za použití pokročilé metody Mixed strategies Differential evolution (MSDE) bylo pro lokalizaci minima s přesností na 3 desetinná místa průměrně potřeba 2400 simulací. Při užití starší metody Differential evolution (DE) pak bylo pro lokalizaci minima s přesností na 6 desetinných míst potřeba 7190 simulací (testování proběhlo v 30 optimalizačních pokusech). Desetirozměrný problém Ackley's function zvládla dle [39] metoda MSDE s průměrným počtem 14350 simulací s přesností na 7 desetinných míst. Metoda DE pak stejný problém řešila s využitím průměrně 31040 simulací s přesností na 6 desetinných míst.

Metoda AMS se zdá být velice účinným optimalizačním algoritmem pro řešení problémů nižších dimenzí. Její efektivita při řešení mnohodimenzionálních problémů bude předmětem dalšího testování. Doposud provedené optimalizační testy naznačují, že algoritmus AMS nastavený dle vztahu (29) je při řešení prozatím testovaných problémů účinnější, než metoda Simulovaného žíhání. Další srovnání účinnosti algoritmu AMS a jiných konvenčních optimalizačních postupů bude jedním z cílů budoucího testování této metody.

4. PŘÍKLADY VYUŽITÍ PROGRAMU FNPO

Obecný popis úloh řešitelných pomocí programu FNPO je uveden v kapitole 2.2.2. Podrobný popis práce s programem je součástí uživatelského manuálu v příloze P1. Následující kapitola se bude zabývat spolehlivostní optimalizací prováděnou na praktických příkladech pomocí akademického softwaru FNPO.

4.1 PŘÍKLAD VYUŽITÍ FNPO K NEANALYTICKÉMU ŘEŠENÍ ROVNIC

Díky implementaci stochastické optimalizační metody AMS v programu FNPO je možné jej využít k neanalytickému řešení rovnic a složitých integrálů. Existuje mnoho typů rovnic, jejichž analytické řešení není možné. Příkladem jednoduché rovnice tohoto typu může být vztah (73):

$$a = \frac{\cos(x)}{x} \tag{73}$$

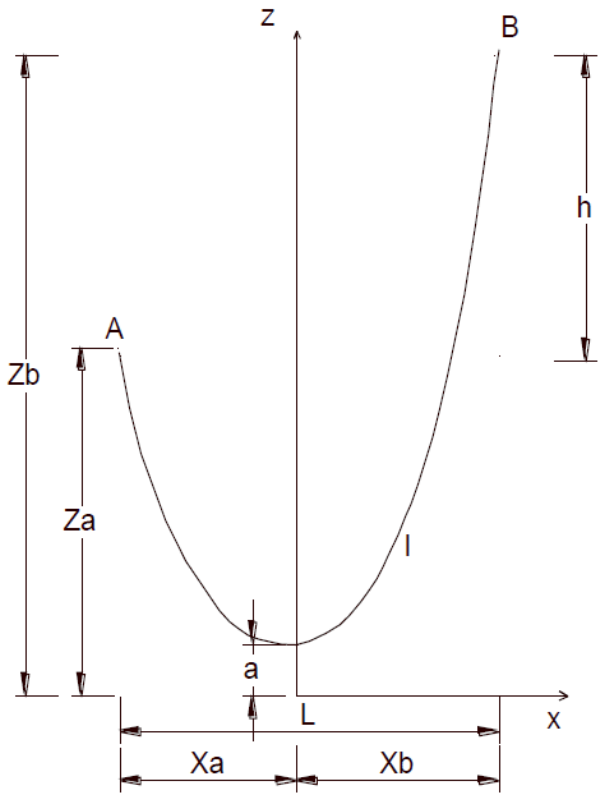
kde a je libovolné reálné číslo. Tyto vztahy se často objevují při řešení různých typů diferenciálních rovnic. Způsob využití FNPO k řešení těchto problémů odpovídá definici optimalizační úlohy dle vztahu (60). Následující řešený příklad byl převzat z [60] a demonstruje využití FNPO ke stanovení reálného parametru a rovnice tížné řetězovky. S využitím infinitesimálního elementu délky lana ds byla v [61] odvozena diferenciální rovnice pravé tížné řetězovky:

$$\frac{d^2z}{dx^2} = \frac{qds}{N_0 dx} \tag{74}$$

kde q je hodnota rovnoměrného zatížení řetězovky a N_0 je hodnota vodorovné složky tahové síly. Uvážíme-li popis geometrie lana zobrazený na Obr. 39, vede řešení diferenciální rovnice (74) a postupná úprava získaných vztahů na rovnici pravé tížné řetězovky v explicitním tvaru:

$$z = a\,cosh\frac{x}{a} \tag{75}$$

kde x a z jsou souřadnice jednotlivých bodů zavěšeného lana délky l.



Obr. 39 Popis geometrie lana pro odvození vztahu (75) – převzato z [60]

Délku řetězovky l lze vypočíst dle následujícího integrálního vztahu:

$$l = \int_A^B \sqrt{1 + \left(\frac{dz}{dx}\right)^2}\, dx = \left[asinh\frac{x}{a}\right]_A^B \tag{76}$$

Podrobný popis odvození výše uvedených vztahů je uveden v [60], [61]. Nyní máme k dispozici všechny základní vztahy potřebné pro nalezení tvaru, který zaujme neprotažitelné lano zavěšené mezi dvěma body. Formulujme nyní následující úlohu:

Jsou dány 2 body (A, B) vzdálené 80 m. Rozdíl jejich výšek je 10 m (bod B je výše). Chceme je spojit lanem délky 85 m. Jaký tvar zaujme lano, bude-li:

- dokonale tuhé (tj. E = ∞ GPa)
- L = 80 m
- l = 85 m
- h (rozdíl výšek) = 10 m
- q = 41,1 N/m
- A = 0,0009898 m²

Geometrie zadané úlohy odpovídá Obr. 39.

Odvození rovnice řetězovky pro zadanou úlohu je uvedeno v [60] a vede na vztah (77), který není řešitelný analyticky.

$$\sqrt{l^2 - h^2} = 2asinh\frac{L}{2a} \tag{77}$$

Jedná se tedy o rovnici o jedné neznámé proměnné *a*. Po získání hodnoty *a* je možné snadno dopočíst polohu libovolného bodu křivky prověšení lana s využitím vztahu (75). Uvedený problém je velice jednoduchý a lze jej snadno řešit např. metodou prosté iterace. Cílem této úlohy je však demonstrovat na jednoduchém příkladu postup využití programu FNPO pro řešení úloh tohoto typu, které mohou být obecně složité např. s řadou lokálních extrémů.

Nejprve je třeba formulovat problém jako optimalizační úlohu danou vztahem (60). Pravou stranu vztahu (77) definujeme v programu FReET jako funkci jediné neznámé náhodné proměnné *a* definované rovnoměrným rozdělením pravděpodobnosti (z důvodu rovnoměrného pokrytí návrhového prostoru). Rozsah hodnot proměnné *a* stanovíme např. (-300, 300). Algoritmus AMS konverguje (s určitou pravděpodobností) i k řešením problému, která se mohou nacházet mimo původně zadanou oblast. Vzhledem k jednoduchosti cílové funkce je možné pro řešení využít vyššího celkového počtu simulací (např. 5000). Poté načteme definovaný .fre soubor v programu FNPO a nastavíme parametry algoritmu AMS se zohledněním vztahu (29). V sekci "Optimize to" zatrhneme volbu "Close to:" a jako cílovou

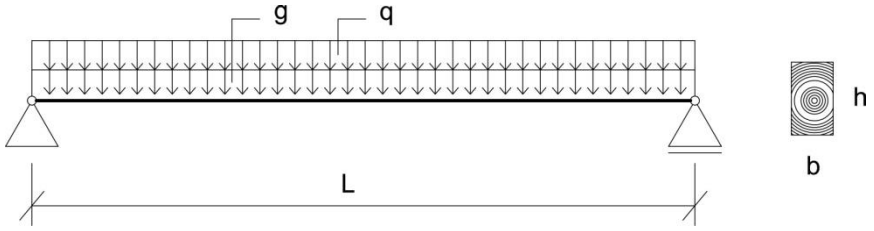
funkční hodnotu nastavíme vyčíslenou hodnotu levé strany vztahu (77). Aplikací tohoto postupu byla získána hledaná hodnota reálného parametru $a=70,12224$ - ta odpovídá hodnotě řešení uvedeného v [60] ($a=70,1222$). Podobným způsobem lze postupovat i při řešení složitějších problémů tohoto typu.

Uvedený příklad popisuje spíše matematickou aplikaci algoritmu AMS. Úloha je svou podstatou podobná optimalizačním problémům uvedeným v kapitole 3.2. Následující příklady využití FNPO se proto zaměří na spolehlivostní optimalizaci jednoduchých problémů běžných ve stavební praxi.

4.2 MINIMALIZACE PRŮŘEZOVÉ PLOCHY SE SPOLEHLIVOSTNÍM OMEZENÍM

Pro potřeby úlohy 4.2 a 4.3 definujme nyní problém převzatý z [62]. Jedná se o úlohu spolehlivostní optimalizace dřevěného nosníku. Analytické vztahy potřebné k definici užitých funkcí mezního stavu jsou převzaty z [53].

Dřevěný prostě podepřený nosník délky l obdélníkového průřezu je zatížen rovnoměrným zatížením po celé své délce. Toto zatížení je součtem stálého zatížení g nahodilého zatížení q. Statické schéma je zobrazeno na Obr. 40.



Obr. 40 Statické schéma prostě podepřeného dřevěného nosníku s obdélníkovým průřezem

Parametry b a h představují výšku a šířku průřezu. Pro potřeby spolehlivostní optimalizace v souladu s požadavky EUROCODE 5 [53] definujme dvě základní funkce mezního stavu. Pro mezní stav únosnosti:

$$g_1 = M_R - M_E \tag{78}$$

a pro mezní stav použitelnosti:

$$g_2 = u_{lim,fin} - u_{net,fin} \tag{79}$$

kde M_R je moment na mezi únosnosti průřezu daný vztahem (80).

$$M_R = \Theta_R \frac{1}{6} b h^2 k_{mod} f_m \tag{80}$$

M_E je moment vyvolaný působícím zatížením:

$$M_E = \Theta_E \frac{1}{8} (g + q) l^2 \tag{81}$$

Ve vztazích (80) a (81) představují hodnoty Θ_R a Θ_E modelované nejistoty odezvy konstrukce a účinku zatížení. k_{mod} je normovým součinitelem zohledňujícím vliv okolní vlhkosti a doby trvání zatížení. Hodnota f_m představuje ohybovou pevnost použitého dřeva. V případě vztahu (79) je funkce mezního stavu definována pomocí přetvoření. Mezní hodnota přetvoření je dána vztahem:

$$u_{lim,fin} = \frac{l}{200} \tag{82}$$

Hodnota přetvoření vlivem působícího zatížení je definována takto:

$$u_{net,fin} = \Theta_E (u_{1,fin} + u_{2,fin}) \tag{83}$$

kde:

$$u_{1,fin} = \frac{5}{384} \frac{g l^4}{E \frac{1}{12} b h^3} (1 + k_{1,def}) \tag{84}$$

$$u_{2,fin} = \frac{5}{384} \frac{q l^4}{E \frac{1}{12} b h^3} (1 + k_{2,def}) \tag{85}$$

$u_{1,fin}$ a $u_{2,fin}$ jsou přetvoření vyvolaná stálým a nahodilým zatížením, E je modul pružnosti v tahu použitého dřeva, $k_{1,def}$ je faktor zohledňující vliv dotvarování pro stálé zatížení, $k_{2,def}$ je stejným faktorem pro zatížení nahodilé. V prováděných výpočtech byly uvažovány tyto hodnoty normových součinitelů:

- $k_{mod} = 0{,}8$
- $k_{1,def} = 0{,}8$
- $k_{2,def} = 0{,}25$

Spolehlivostní i optimalizační proces vyžaduje randomizaci jednotlivých parametrů vektoru vstupních hodnot. Pro potřeby výpočtu indexu spolehlivosti metodou FORM v rámci vnitřního cyklu spolehlivostní optimalizace bylo znáhodnění jednotlivých parametrů provedeno dle tabulky 11 [62].

Tab. 11 Randomizace parametrů pro spolehlivostní výpočty

Proměnná	Rozdělení	střední hodnota	Směrodatná odchylka	Variační koeficient
l [m]	Normální	3.5	0,175	0,05
b [m]	Normální	?	--	0,05
h [m]	Normální	?	--	0,05
E [Gpa]	Lognormální (2 par)	10	1,3	0,13
f_m [Mpa]	Lognormální (2 par)	34	8,5	0,25
g [kN/m]	Gumbelovo max EV 1	1,686	0,169	0,1
q [kN/m]	Gumbelovo max EV 2	2,565	0,77	0,3
Θ_R [-]	Lognormální (2 par)	1	0,1	0,1
Θ_E [-]	Lognormální (2 par)	1	0,1	0,1

Střední hodnoty výšky a šířky průřezu jsou předmětem optimalizace. Představená úloha je tedy devítidimensionální z pohledu spolehlivostních výpočtů a dvojdimensionální z pohledu optimalizačního procesu. Pro potřeby optimalizace průřezové plochy byly parametry b a h randomizovány dle tabulky 12.

Tab. 12 Randomizace parametrů b a h pro potřeby optimalizace průřezové plochy

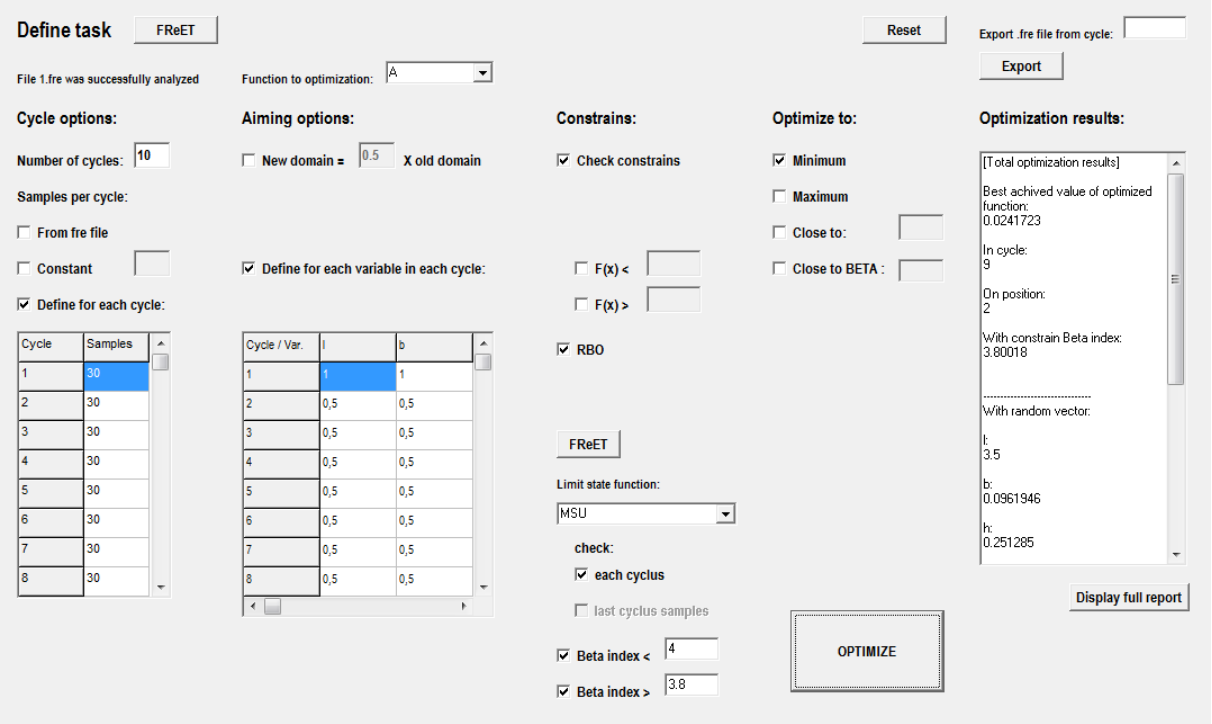
Proměnná	Rozdělení	střední hodnota	Směrodatná odchylka	a	b
b [m]	Rovnoměrné	0,125	0,0144	0,1	0,15
h [m]	Rovnoměrné	0,225	0,0144	0,2	0,25

Formulujme nyní optimalizační úlohu, v níž cílem optimalizace bude nalezení minimální možné průřezové plochy představeného dřevěného nosníku za předpokladu spolehlivostního omezení daného funkcemi mezního stavu definovanými vztahy (78) a (79). Jedná se tedy o definici úlohy spolehlivostní optimalizace odpovídající vztahu (59), s omezením daným vztahem $d < \beta_o(\boldsymbol{X})$, kde d odpovídá minimální normou stanovené hodnotě indexu spolehlivosti. Normou povolená hodnota indexu spolehlivosti pro mezní stav únosnosti (MSU) je 3,8. Pro mezní stav použitelnosti (MSP) předepisuje EUROCODE minimální hodnotu indexu spolehlivosti 1,5. Tyto hodnoty budou použity jako omezující podmínky optimalizace.

Pro řešení definované úlohy spolehlivostní optimalizace s pomocí programu FNPO je nutné nadefinovat 2 zdrojové .fre soubory pomocí programu FReET. První ze souborů definuje úlohu ve vnějším (optimalizačním cyklu). V rámci tohoto souboru jsou definovány pouze 2 náhodné proměnné dle tabulky 12. Zbylé návrhové veličiny jsou uvažovány jako deterministické definované svými středními hodnotami. Tento soubor také musí obsahovat cílovou funkci, jež má být minimalizována (v našem případě tedy $A=b \times h$). Druhý ze souborů definuje spolehlivostní úlohu v rámci vnitřního cyklu. Musí obsahovat funkce mezního stavu dané vztahy (78) a (79). Náhodné proměnné jsou v tomto souboru definovány

dle tabulky 11. Na místo středních hodnot proměnných *b* a *h* jsou během procesu dosazovány hodnoty generované ve vnějším optimalizačním cyklu. Program FNPO zatím neumožňuje definici více než jedné omezující podmínky. Danou úlohu proto bude třeba nejprve vyřešit s omezující podmínkou danou funkcí mezního stavu pro mezní stav únosnosti a poté znovu s omezující podmínkou odpovídající meznímu stavu použitelnosti. U obou řešení poté vypočteme (v programu FReET, pomocí .fre souboru generovaného u nejlepšího vzorku pro spolehlivostní výpočet) také index spolehlivosti druhé definované funkce mezního stavu. Jako finální řešení bude voleno to, které bude mít menší výslednou průřezovou plochu a zároveň splní normové požadavky pro oba mezní stavy.

Po definici obou zdrojových .fre souborů byl nastaven algoritmus AMS na celkový počet 300 simulací realizovaných v 10 cyklech s hodnotou q=0,5. Jako omezující podmínka byla nejprve stanovena minimální normou požadovaná hodnota indexu spolehlivosti pro funkci limitního stavu danou vztahem (78) tedy 3,8. Výsledné řešení této úlohy je vidět na Obr. 41.



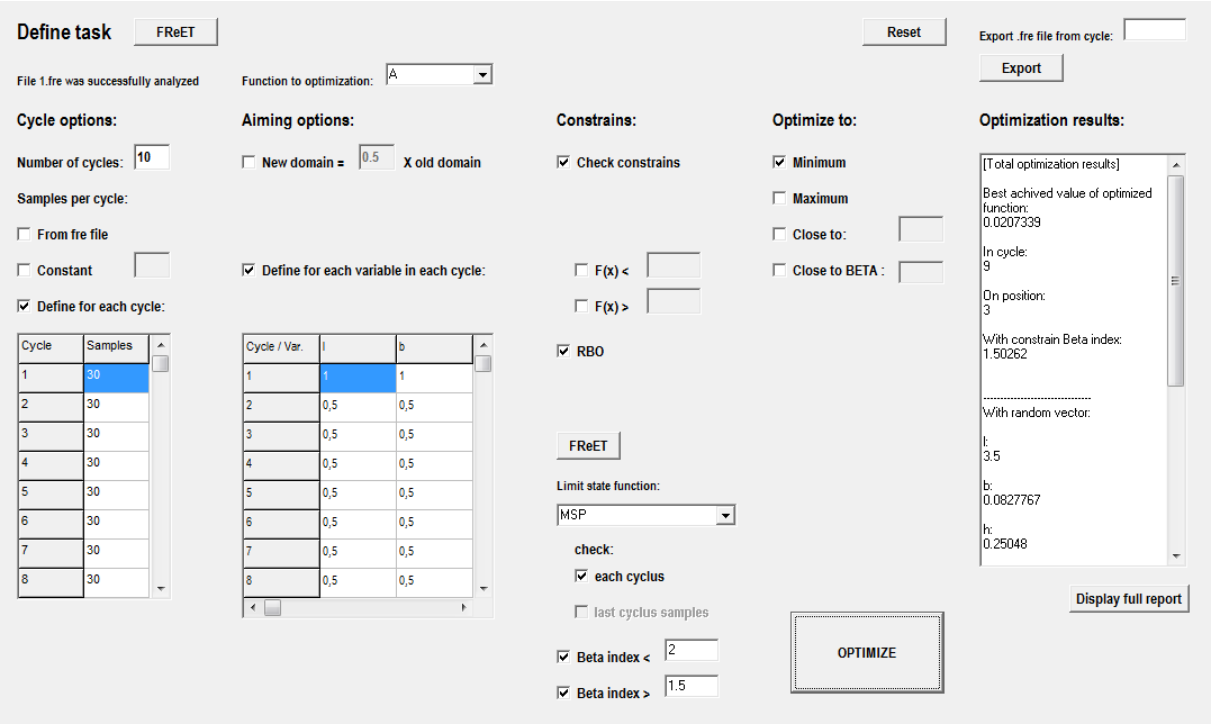
Obr. 41 Kopie obrazovky s výsledným řešením pro omezení dané MSU

Výsledek spolehlivostní optimalizace tedy vypadal takto:

A = 0.0241723 m^2
Pro náhodný vektor:
b = 0.0961946 m, h = 0.251285 m
S hodnotami indexu spolehlivosti:
pro MSU β =3.80018 a pro MSP β = 2.013

Oba normové požadavky tedy byly splněny.

Následně byl stejný problém řešen s využitím omezující podmínky definované pomocí druhé funkce mezního stavu (MSP) s minimální dovolenou hodnotou indexu spolehlivosti 1,5 (Obr. 42).



Obr. 42 Kopie obrazovky s výsledným řešením pro omezení dané MSP

Výsledek spolehlivostní optimalizace:

A = 0.0207339 m^2
Pro náhodný vektor:
b = 0.0827767 m, h = 0.25048 m
S hodnotami indexu spolehlivosti:
pro MSU $\beta = 3.3901$ a pro MSP $\beta = 1.50262$

V druhém případě sice bylo dosaženo menší hodnoty průřezové plochy, nebyl však splněn normový požadavek na minimální hodnotu indexu spolehlivosti 3,8 pro MSU. Za výsledné řešení tedy můžeme považovat výstup z testu provedeného za použití spolehlivostní omezující podmínky dané funkcí mezního stavu pro MSU.

Vzhledem k podobě vztahů (80), (84) a (85) v nichž figuruje hodnota h ve vyšší mocnině, než hodnota b se dalo očekávat, že výsledkem optimalizace budou nejspíše štíhlé vysoké tvary průřezu. Jako optimální se může jevit, pokusit se navrhnout průřez tak, aby hodnota indexu spolehlivosti byla na hranici normových požadavků jak pro MSU, tak pro

MSP. K vypracování takového návrhu je potřeba provést multikriteriální optimalizaci s cílem dosažení požadované kombinace hodnot indexů spolehlivosti obou funkcí mezního stavu. Implementace možnosti provádět multikriteriální optimalizaci je plánována až do další verze programu FNPO. Program však umožňuje použít index spolehlivosti jedné z funkcí mezního stavu jako cílovou hodnotu optimalizace zatímco druhá z požadovaných hodnot indexu spolehlivosti může definovat spolehlivostní omezení. Takovýto způsob řešení "dvoukriteriální" optimalizační úlohy popisuje následující kapitola.

4.3 CÍLENÉ HLEDÁNÍ VSTUPNÍCH HODNOT PŘI ZADANÉ VÝSTUPNÍ ÚROVNI SPOLEHLIVOSTI

Příklad popsaný v kapitole 4.2 byl v [62] řešen pomocí neuronové sítě. Cílem úlohy bylo nalézt takovou kombinaci výšky a šířky průřezu, aby hodnota indexu spolehlivosti pro MSU byla 3,8 a pro MSP 1,5. Výsledek řešení zmíněného problému pomocí neuronové sítě uvedený v [62] je zapsán v tabulce 13.

Tab. 13 Výsledek cíleného hledání vstupních hodnot při zadané výstupní úrovni spolehlivosti získaný pomocí neuronové sítě [62]

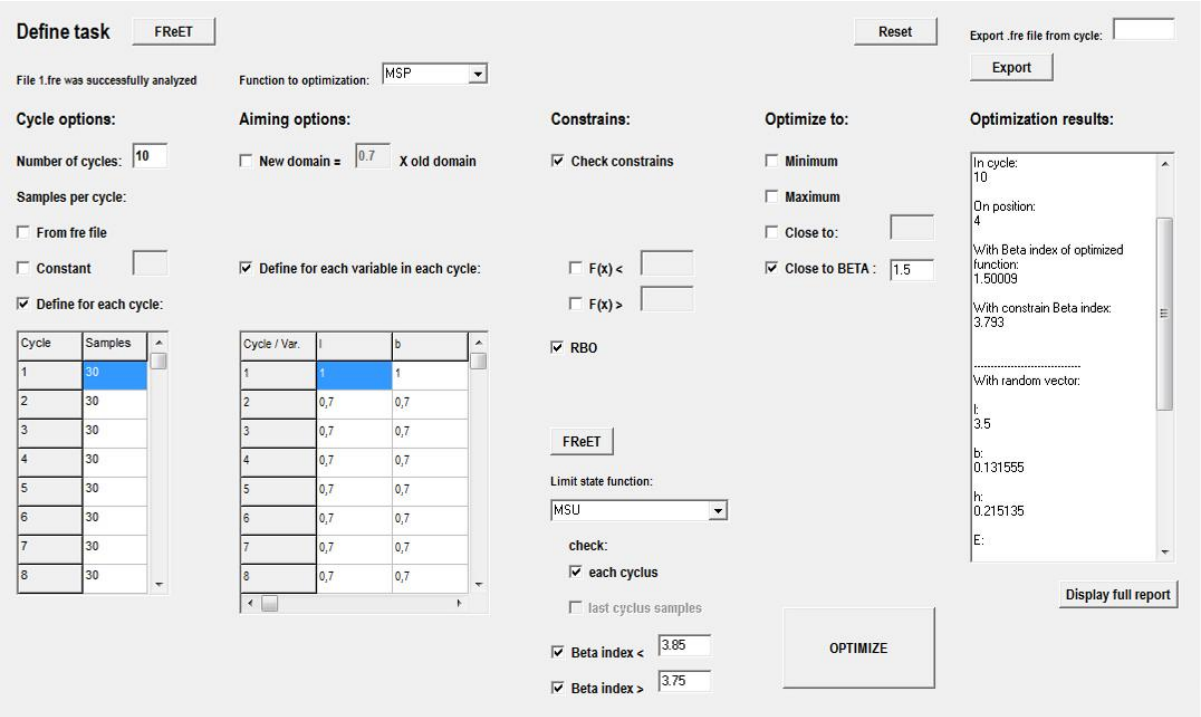
střední hodnota h	střední hodnota b	β MSU	β MSP	β MSU-cílová	β MSP-cílová
0,13244	0,21432	3,8001	1,5001	3,8	1,5

Definujeme-li optimalizační úlohu dle vztahu (61) za použití spolehlivostní omezující podmínky dané vztahem (66), pak můžeme postupy spolehlivostní optimalizace v programu FNPO využít k řešení stejného problému. Vzhledem k tomu, že uvedený postup řešení není multikriteriální optimalizací v pravém slova smyslu, není možné očekávat dosažení podobné přesnosti jako v případě použití neuronové sítě. Budoucí implementace multikriteriální optimalizace spolu s rozšířením možností definice omezujících podmínek by mohla programu FNPO umožnit řešit zmíněné problémy s vyšší přesností.

Během řešení úlohy programem FNPO byla využita možnost stanovit cílovou hodnotu indexu spolehlivosti pro vybranou funkci mezního stavu. Byla tedy definována cílová hodnota indexu spolehlivosti pro funkci mezního stavu danou vztahem (81) $\beta = 1{,}5$. Jako omezující podmínka byl stanoven interval přípustných hodnot indexu spolehlivosti pro funkci poruchy danou vztahem (80) $3{,}75 < \beta < 3{,}85$. Jako zdrojové .fre soubory bylo možné použít tytéž soubory, které byly využity při řešení úlohy minimalizace průřezové plochy v předchozí kapitole. Randomizace proměnných pro spolehlivostní cykly probíhala opět dle tabulky 11, pro účely optimalizace byly proměnné randomizovány dle tabulky 12.

Při řešení úlohy za použití algoritmu AMS bylo opět užito celkového počtu 300 simulací. Optimalizaci jednoduché dvojdimenzionální úlohy by algoritmus AMS pravděpodobně zvládl i s nižším počtem simulací, vzhledem ke specifikaci úlohy však bylo

nutné zajistit, aby přísné definici omezující podmínky vyhověla alespoň jedna realizace v prvním optimalizačním cyklu algoritmu AMS. Tento problém by odstranila aplikace postupů multikriteriální optimalizace. Výsledek řešení uvedené úlohy pomocí programu FNPO je viditelný na Obr. 43.



Obr. 43 Výsledné řešení zadané úlohy v programu FNPO

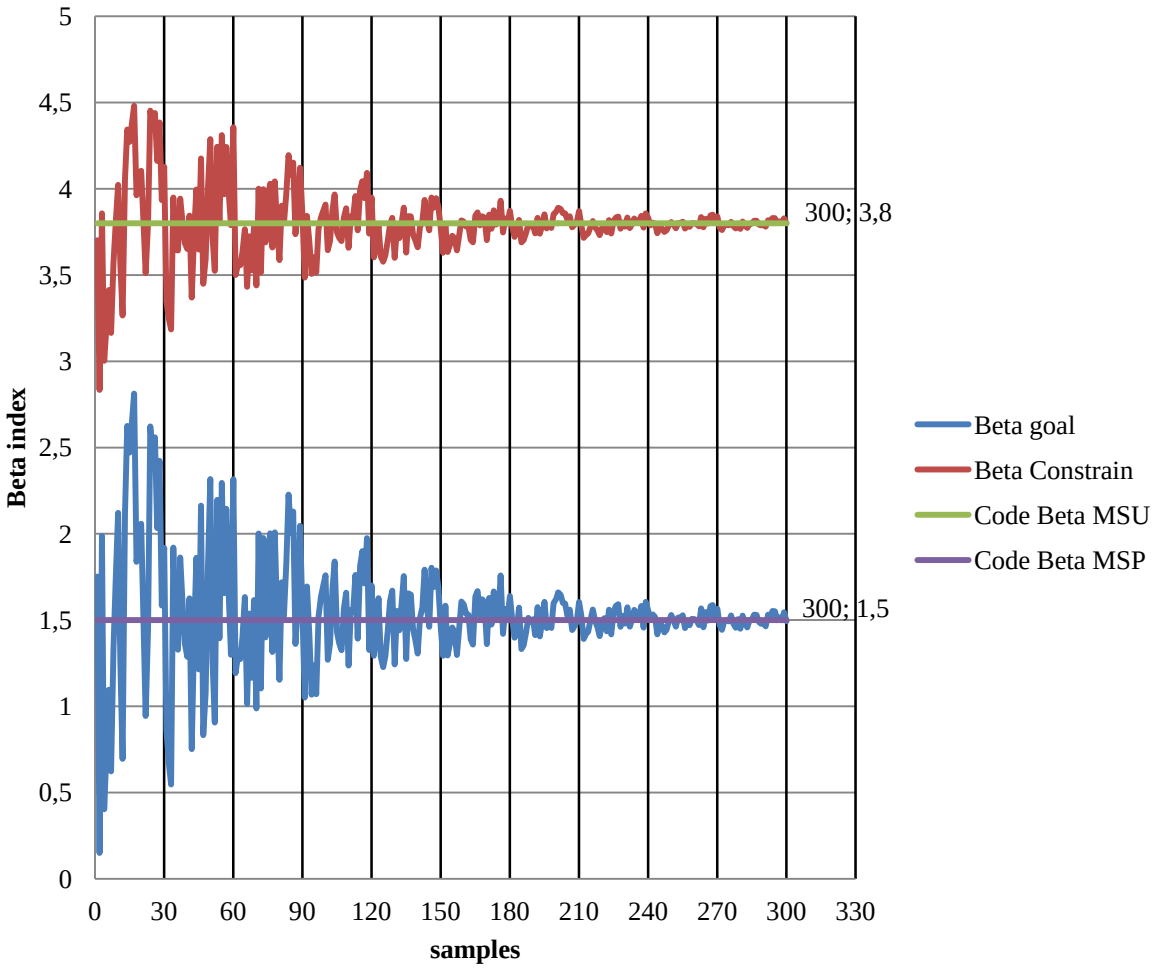
Výsledek úlohy identifikace vstupních parametrů se zadanou výstupní úrovní spolehlivosti:

Dosažené hodnoty indexů spolehlivosti:
pro MSU $\beta = 3.793$ a pro MSP $\beta = 1.50009$

Pro návrhový vektor:
$b = 0{,}1311555$, $h = 0{,}215135$

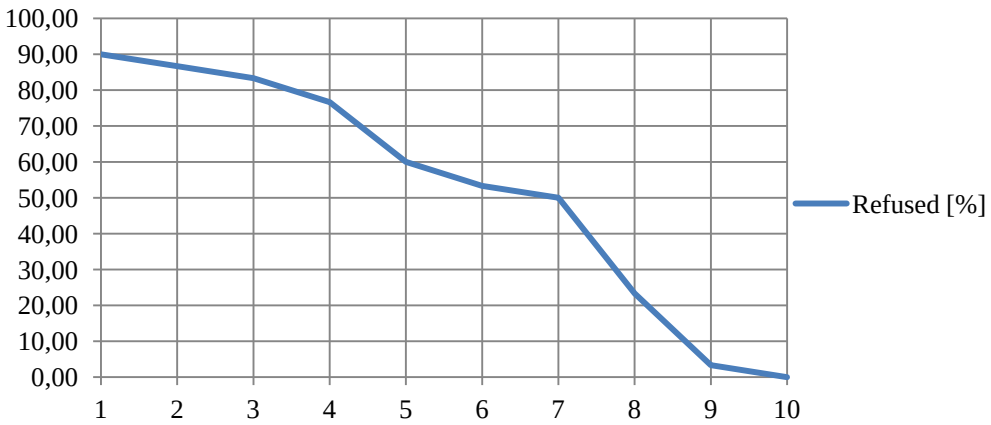
Nalezené řešení tedy přibližně odpovídá hodnotám získaným pomocí neuronové sítě [62]. Získaná průřezová plocha má velikost 0,028302 m^2. Je tedy větší než plocha získaná přímou minimalizací funkce plochy v příkladu uvedeném v kapitole 4.2. Mohlo by se tedy zdát, že uvedené řešení je náročnější a přesto méně ekonomické. Definované funkce mezního stavu však nezohledňují vliv štíhlosti průřezu a případné boulení. Štíhlé průřezy získané přímou minimalizací průřezové plochy by při zavedení robustnějšího výpočetního modelu pravděpodobně nevyhověly spolehlivostním požadavkům.

Graf 15 zobrazuje postupnou konvergenci generovaných řešení směrem k požadovaným hodnotám indexů spolehlivosti pro případ spolehlivostní optimalizace odpovídající Obr. 43.



Graf 15 Vývoj indexů spolehlivosti v průběhu řešení popsané úlohy

Graf 16 zachycuje postupný vývoj procentuálního množství realizací, jež nevyhověly omezujícím podmínkám optimalizace.



Graf 16 Vývoj množství zamítnutých realizací v průběhu optimalizačního procesu

ZÁVĚR

V první části této práce byl popsán současný stav poznání v oboru spolehlivostní optimalizace konstrukcí. Kapitola 1. přináší přehled dostupných matematických metod a prostředků používaných běžně pro spolehlivostní optimalizaci konstrukcí. Byla představena zcela nová optimalizační metoda nazvaná Aimed Multilevel Sampling (AMS), která využívá víceúrovňové simulace některou s dostupných simulačních metod a postupného soustředění generovaných vzorků náhodného prostoru do oblasti s nejlepšími hodnotami cílové funkce. Pro tuto metodu byly v rámci této práce představeny obecné principy jejího využívání a byly odvozeny vztahy, na jejichž základě je možné nově navržený algoritmus AMS nastavit tak, aby poskytoval co nejlepší výsledky při optimalizaci neznámé funkce se zvoleným konečným počtem použitých vzorků. Kapitola 2. popisuje software FReET Nested Probabilistic Optimizer (FNPO) vyvinutý autorem pro účely spolehlivostní optimalizace. Tento program využívá metody AMS v rámci optimalizačního cyklu procesu spolehlivostní optimalizace. FNPO umožňuje spolehlivostní optimalizaci funkcí obecné složitosti s využitím cyklického spouštění programu FReET a úpravy jeho vstupních a výstupních souborů. V kapitole 3. byly představeny dosavadní výsledky testování účinnosti algoritmu AMS na běžných testovacích optimalizačních problémech. Bylo také provedeno několik testů srovnávajících účinnost AMS a metody Simulovaného žíhání. Provedené testy prokázaly schopnost algoritmu AMS nalézt globální extrém funkce u problémů jedné až deseti dimenzí. Pro přesné stanovení účinnosti navrženého algoritmu bude nutné provést další testování na jiných běžně užívaných optimalizačních příkladech. Výsledky předběžných testů však naznačují, že metoda AMS by mohla být účinným nástrojem pro analýzu s malým počtem vzorků. V kapitole 4. byly prezentovány ukázky spolehlivostní optimalizace praktických problémů s využitím programu FNPO. V tomto textu prezentované techniky a postupy je možné aplikovat na problémy spolehlivostní optimalizace libovolných modelů, včetně složitých výpočetně náročných úloh mechaniky kontinua. Další výzkum se bude ubírat tímto směrem.

POUŽITÉ INFORMAČNÍ ZDROJE

[1] TEPLÝ, B. a D. NOVÁK. *Spolehlivost stavebních konstrukcí: teorie, numerické metody, navrhování, software*. 1. vyd. Brno: CERM, 1999, 87 s. ISBN 80-214-1149-X.

[2] DEOLALIKAR, V. *P ≠ NP*. HP Research Labs, Palo Alto. 2010, s. 50. Dostupné z: http://www.scribd.com/doc/35539144/pnp12pt

[3] ČERNÝ, V. *Thermodynamical approach to the traveling salesman problem: An efficient simulation algorithm*. Journal of Optimization Theory and Applications. 1985, vol. 45, issue 1, s. 41-51. DOI: 10.1007/BF00940812. Dostupné z: http://link.springer.com/10.1007/BF00940812

[4] HASOFER, A. M. a N. C. LIND. *Exact and invariant second-moment code format.* Journal of Eng. Mech. Division. Vol. 100. ASCE, 1974.

[5] FREUDENTHAL, A. M. *Safety and the probability of structural failure*. Transactions ASCE. 1956, s. 60.

[6] FREUDENTHAL, A. M., J. M. GARRELTS a M. SHINOZUKA. *The analysis of structural safety*. Journal of the Structural Division ASCE. 1966, s. 58.

[7] FREUDENTHAL, A. M. a W. S. GITHER. *Probabilistic approach to economic design of marine structures.* In: 22nd International Navigation Congress. 1969, s. 14.

[8] FREUDENTHAL, A. M. *Safety and reliability of large engineering structures*. In: National Academy of Sciences: Public Safety: Growing Factor in Modern Design. 1970, s. 5.

[9] SLOWIK, O. *Optimalizace betonových konstrukcí stochastickými metodami optimalizace*. Brno, 2012. Dostupné z: https://dspace.vutbr.cz/handle/11012/17001. Bakalářská práce. VUT Brno. Vedoucí práce prof. Ing. Drahomír Novák, DrSc.

[10] FRANGOPOL, D. M. *Reliability-based structural optimization research at the University of Colorado: A brief retrospective 1983-1991*. Progress in Structural Engineering. 1991, s. 21.

[11] FRANGOPOL, D. M. a M. IIZUKA. *Multiobjective decision support spaces for optimum design of nondeterministic structural systems*. Probabilistic Safety Assessment and Management, Elseviver. 1991, s. 5.

[12] KUPFER, H. a A. M. FREUDENTHAL. *Structural Optimization and risk control*. In: ICOSSAR'77 (2nd Intern. Conf. on Structural Safety and Reliability, Sept. 19-21, München, Germany 1977). Düsseldorf: Werner-Verlag, 1977, s. 12.

[13] JONES, N. K. W. *Risk-based optimization of maintenance - methods and approaches*. In: European Safety and Reliability Conference ESREL'93. Munich, Germany, 1993, s. 18.

[14] YANG, J-N. *Application of reliability methods to fatigue, quality assurance and maintenance*. In: SCHUËLLER, G. I. ICOSSAR'93 (6th Intern. Conf. on Structural

Safety and Reliability, Innsbruck, Austria, Aug. 9-13 1993). Rotterdam, The Netherlands: Balkema, 1994, s. 15.

[15] STRANG, G. a G. J. FIX. *An analysis of the finite element method*. Englewood Cliffs, N.J.: Prentice-Hall, 1973, xiv, 306 p. ISBN 01-303-2946-0.

[16] SCHREIDER, Y. A. *The Monte Carlo method: The method of statistical trials*. Oxford: Pergamon Press, 1967.

[17] RUBINSTEIN, R. *Simulation and Monte Carlo method*. Oxford: John Wiley & Sons, New Press, 1981.

[18] BOURGUND, U. a C. G. BUCHER. *A Code for importance sampling procedure using design points – ISPUD: A user manual*. Inst. Eng. Mech., Innsbruck University, Report No. 8, 1986.

[19] BUCHER, C. G. *Adaptive sampling: an iterative fast Monte-Carlo procedure.* J. Structural safety. Vol. 5, No. 2, 1988.

[20] SCHUËLLER, G. I., C. G. BUCHER, U. BOURGUND a W. OUYPORNPRASERT. *On efficient computationalschemes to calculate structural failure probabilities*. Probabilistic Engineering Mechanics. 1989, s. 8.

[21] SCHUËLLER, G. I. a R. STIX. *A critical appraisal of methods to determine failure probabilities*. Structural Safety. 1987, s. 58.

[22] AYYUB, B. M. a A. HALDAR. *Practical structural reliability techniques*. Journal of structural engineering, 1984.

[23] MCKAY, M. D., W. J. CONOVER a R. J. BECKMAN. *A comparison of three methods for selecting values of input variables in the analysis of output form a computer code.* Technometrics, Vol. 21, 1979.

[24] AYYUB, B. M. a K. L. LAI. *Structural reliability assessment using Latin Hypercube Sampling*. In Proc. of ICOSSAR'89, the 5th International conference on structural safety and reliability, San Francisco, USA, Vol 1, Structural safety and reliability , 1989.

[25] BUCHER, C. G. a U. BOURGUND. *Efficient use of Response surface methods*. Inst. Eng. Mech., Innsbruck University, report No. 9, 1987.

[26] GRIGORIU, M. *Methods for approximate reliability analysis.* J. Structural safety. No. 1, 1982/1983.

[27] LI, K. S. a P. LUMB. *Reliability analysis by numerical integration and curve fitting.* J. Structural safety. Vol. 3, 1985.

[28] ČERVENKA CONSULTING LTD. *Www.freet.cz* [online]. 2004 [cit. 2013-11-29]. Dostupné z: http://www.freet.cz/

[29] KIRKPATRICK, S., Jr., C. D. GELATT a M. P. VECCHI. *Optimization by Simulated annealing*. IBM Research report, RC 9355, 1982.

[30] SILBRNÍK, V. *Pravděpodobnostní optimalizace nosných stavebních konstrukcí metodou evolučních strategií*. Brno, 1994. Diplomová práce. VUT Brno. Vedoucí práce Ing. Drahomír Novák, CSc.

[31] RECHENBERG I. *Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution*. Stuttgart-Bad Cannstatt: Frommann-Holzboog, c1973, 170 s. ISBN 37-728-0373-3.

[32] SCHWEFEL, H. P. *Numerical optimization of computer models*. New York: Wiley, c1981, VII, 389 s. ISBN 04-710-9988-0.

[33] POHLHEIM, H. *GEATbx: Example Functions (single and multi-objective functions) 2 Parametric Optimization* [online]. 2006 [cit. 2013-11-29]. Dostupné z: http://www.geatbx.com/docu/fcnindex-01.html

[34] POŠÍK, P. *Paralelní genetické algoritmy*. Praha, 2001. Dostupné z: http://labe.felk.cvut.cz/~posik/dipl/Diplomka.htm. Diplomová práce. ČVUT.

[35] ALI, M., M. PANT, A. ABRAHAM a V. SNAŠEL. *Differential evolution using mixed strategies in competitive environment*. International Journal of Innovative Computing: Information and Control. 2011, roč. 7, č. 8, s. 22.

[36] BUCHER, C. G., H. J. PRADLWARTER a G. I. SCHEUËLLER. *COSSAN - (Computational Stochastic Structural Analysis) - Perspectives of Software Developments*. In: SCHEUËLLER, G. I. ICOSSAR'93 (6th Intern. Conf. on Structural Safety and Reliability, Innsbruck, Austria, Aug. 9-13 1993). Rotterdam, The Netherlands: Balkema, 1994.

[37] COSSAN *Computational Stochastic Structural Analysis, User's manual.* Institute of Engineering Mechnics, University of Insbruck, Austria, 1996.

[38] POSPÍŠILOVÁ, A., E. MYŠÁKOVÁ a M. LEPŠ. *Multi-objective adaptive design of experiments for reliability-based design optimization*. In: NOVÁK, D. a M. VOŘECHOVSKÝ. 11th International Probabilistic Workshop. Brno, Czech Republic: LITERA, 2013, s. 12. ISBN 978-80-214-4800-1.

[39] COHN, M. Z. a A. S. DINOVITZER. *Application of Structural Optimization*. Journal of Structural Engineering. 1994, č. 2, s. 33.

[40] SCHITTKOWSKI, K. *Theory, implementation, and test of a nonlinear programming algorithm*. Euromech-Colloquium 164 on "Optimization methods in structural design", Universität Siegen, 1982, Bibliographisches Institut, Mannheim, 1983, s. 10.

[41] GASSER, M. *Structural optimization based on reliability criteria utilizing approximate methods*. Innsbruck, 1996. Dissertation. Leopold-Franzens-Universität Innsbruck.

[42] MARTI, K. *Stochstic optimization of structural design*. ZAMM – Z. angel. Math. Mech., 1992, s. 12.

[43] ČERNÍK, F. *Ověření výkonnosti metody HSLHS pro odhad statistik náhodných vektorů v úlohách mechaniky*. Brno, 2009. Dostupné z: www.fsv.cvut.cz/svoc/2010/registrd.php?Akce=SHOW&SID=21. Studentská vědecká odborná činnost. VUT Brno. Vedoucí práce doc. Ing. Miroslav Vořechovský, Ph.D.

[44] BUCHER, C. *Asymptotic sampling for high-dimensional reliability analysis*. Probabilistic Engoneering Mechanics, 2009, s. 6.

[45] *11th International Probabilistic Workshop*. Brno, Czech Republic: LITERA, 2013. D. Novák & M. Vořechovský. ISBN 978-80-214-4800-1.

[46] BASU, D. *Role of the sufficiency and likehood principles in simple survey theory*. Sankhya, 1969, s. 13.

[47] THOMPSON, S. K. *Adaptive sampling*. [online]. 1987, s. 3 [cit. 2013-12-05]. Dostupné z: http://www.amstat.org/sections/srms/Proceedings/papers/1987_139.pdf

[48] THOMPSON, S. K. a G. SEBER. *Adaptive sampling*. New York: Wiley, c1996, XI, 265 s. ISBN 04-715-5871-0.

[49] *Reference Guide Open Turns version 0.13.2*. Http://doc.openturns.org/ [online]. 2010 [cit. 2013-12-06]. Dostupné z: http://doc.openturns.org/openturns-0.13.2/doc/html/ReferenceGuide/output/OpenTURNS_ReferenceGuidesu59.html

[50] MELCHERS, R. E. *Structural system reliability assessment using directional simulation*. Structural Safety, 1994, s. 16.

[51] LEPŠ, M. *Úvod do stochastických optimalizačních metod*. ČVUT. Http://klobouk.fsv.cvut.cz/ [online]. 2012 [cit. 2013-12-06]. Dostupné z: http://klobouk.fsv.cvut.cz/~leps/teaching/mmo/prednasky/prednaska04_intro_SA_TA.pdf

[52] MOLGA, M. a C. SMUTNICKI. *Test functions for optimization needs*. 2005, 43 s. Dostupné z: http://www.zsd.ict.pwr.wroc.pl/files/docs/functions.pdf

[53] *Dřevěné konstrukce podle eurokódu 5*. Vyd. 1. Praha: Informační centrum ČKAIT, 2004, 401 s. ISBN 80-867-6913-5.

[54] NOVÁK, D. a J. ELIÁŠ. *Přednášky z předmětu Spolehlivost stavebních konstrukcí*. Brno, 2012.

[55] NOVÁK, D., TEPLÝ, B. a N. SHIRAISHI. *Sensitivity analysis of structures: A review*. Intern. Conf. CIVIL COMP'93, Edimburg, Scotland, August 1993, s. 6.

[56] PROCHÁZKA, D. *Práce s řetězci: Programovací jazyk C++*. In: *www.ui.pefka.mendelu.cz* [online]. 2010 [cit. 2013-11-29]. Dostupné z: https://ui.pefka.mendelu.cz/files/4_retezce_cpp.pdf

[57] PRATA, S. *Mistrovství v C++*. 3. aktualiz. vyd. Překlad Boris Sokol. Brno: Computer Press, 2007, 1119 s. ISBN 978-80-251-1749-1.

[58] SOBOL, T. *Učebnice C++ Builder* [online]. 2001 [cit. 2013-11-29]. Dostupné z: http://builder.tsx.cz/

[59] VANDEKERCKHOVE. J., NYU STERN. *Anneal.m* [online]. 2006 [cit. 2013-12-27]. Dostupné z: http://pages.stern.nyu.edu/~acollard/anneal.m

[60] VOBECKÝ, J. *Analýza řetězovek*. In: *www.mech.fsv.cvut.cz/* [online]. 2012 [cit. 2013-11-29]. Dostupné z: http://mech.fsv.cvut.cz/wiki/images/6/67/SP-PRPE-2013-Vobecky.pdf

[61] KADLČÁK, J. a J. KYTÝR. *Statika stavebních konstrukcí: základy stavební mechaniky, staticky určité prutové konstrukce*. 3. vyd. Brno: VUTIUM, 2010, 349 s. ISBN 978-80-214-3419-6.

[62] NOVÁK, D. a D. LEHKÝ. *An inverse reliability analysis based on stochastic simulation and artificial neural network*. Cape Town: SEMC, 2010.

SEZNAM POUŽITÝCH ZKRATEK A SYMBOLŮ

E	[Pa]	Youngův modul pružnosti v tahu
f_m	[Pa]	Mez pevnosti za ohybu
k_B	[JK^{-1}]	Bolzmannova konstanta
k_{def}	[-]	Normový součinitel zohledňující vliv dotvarování pro dané zatížení
k_{mod}	[-]	Normový součinitel zohledňující vliv vlhkosti a doby trvání zatížení
M_E	[Nm]	Moment vyvolaný působením zatížení
M_R	[Nm]	Moment na mezi únosnosti průřezu
p_f	[%]	Pravděpodobnost poruchy
$u_{lim,fin}$	[m]	Mezní dovolená hodnota průhybu
$u_{net,fin}$	[-]	Průhyb vyvolaný působícím zatížením
β	[-]	Index spolehlivosti

SEZNAM PŘÍLOH

P1 – Uživatelský manuál k programu FReET Nested Probabilistic Optimizer

PŘÍLOHA P1

UŽIVATELSKÝ MANUÁL PRO FREET NESTED PROBABILISTIC OPTIMIZER

FReET Nested Probabilistic Optimizer (FNPO) je program tvořený 3610 řádky zdrojového kódu zapsaného v jazyce C++ [57]. Při práci na programu byl využit překladač Borland C++ Builder 6 [58]. FNPO je 32bitovou aplikací pracující v operačním systému Windows (XP a vyšším). FNPO vyžaduje ke svému fungování program FReET [28]. Obsluha programu FReET je popsána v jeho manuálech. Návod na obsluhu FReETu proto není součástí tohoto textu a při dále uvedeném popisu práce s FNPO se předpokládá základní uživatelská znalost práce s programem FReET. FNPO je akademickým softwarem a nepřepokládá se jeho masové využití mimo akademickou půdu v jeho stávající verzi.

1. ZÁKLADNÍ PRINCIPY

Program FReET umožňuje modelovat nejrůznější stochastické problémy. Je možné v něm definovat libovolnou funkci pomocí equation editoru či DLL. Vstupním proměnným se ve FReETu dá přiřadit jedno z předefinovaných rozdělení pravděpodobnosti včetně definice jeho parametrů (střední hodnota, směrodatná odchylka, šikmost, špičatost). Dále FReET obsahuje několik technik pro simulaci v rámci návrhového prostoru daného problému a dokáže provádět výpočty spolehlivosti metodou FORM. Kromě zmíněných funkcí má program FReET také další jiné možnosti využití blíže v [28]. Úlohy definované ve FReEtu je možné ukládat do jednoduše editovatelných textových souborů. FReET tedy poskytuje vhodný výpočetní základ pro spolehlivostní optimalizační problémy.

Základní myšlenkou motivující k tvorbě FNPO bylo využít program FReET jako výpočetní základ pro obslužný program umožňující stochastickou optimalizaci s využitím nově navrženého optimalizačního algoritmu AMS (viz kapitola 1.4.3), jehož testování bylo jedním z účelů nově vyvinutého programu. Vytvořený program měl být schopen provádět optimalizaci problémů pomocí metody AMS za zohlednění definovaných omezujících podmínek, jež měly zahrnovat i podmínky spolehlivostní.

Vyvinutý program FNPO tedy pracuje s .fre soubory, které obsahují zdrojová data problémů definovaných v programu FReET. Tyto soubory prochází, lokalizuje v nich potřebná data, která vhodně upravuje a vrací je FReETu zpět k novému vyhodnocení. Umožňuje tak FReETu práci v cyklech. Během optimalizačního cyklu program spustí simulaci pomocí předefinovaného souboru 1.fre. Po provedení simulace je identifikován nejlepší vzorek a tento je konfrontován s omezující podmínkou danou buď funkční hodnotu obecně definované funkce v rámci zdrojového souboru 1.fre, nebo omezením hodnoty indexu spolehlivosti vypočteného FReETem pomocí .fre souboru, který je pro daný vzorek (vektor vstupních hodnot) sestaven na základě jiného speciálního souboru 2.fre, který definuje úlohu ve vnitřním cyklu pro výpočet spolehlivosti. V případě, že řešení neodporuje omezujícím podmínkám, je přijato jako vektor středních hodnot pro simulaci v dalším cyklu algoritmu

AMS. Následně je provedeno omezení velikosti návrhového prostoru násobením směrodatné odchylky jednotlivých proměnných hodnotami q stanovenými v obecném případě pro každou proměnnou v každém cyklu zvlášť. Na základě získaných dat je sestaven nový soubor 1.fre pro simulaci na následující úrovni algoritmu AMS. Celý proces pokračuje, dokud není dosaženo stanovaného počtu úrovní. Je také možné definovat konkrétní hodnotu indexu spolehlivosti, které má být pro danou funkci mezního stavu dosaženo. V takovém případě program provede výpočet indexu spolehlivosti zadané funkce mezního stavu pro každý vygenerovaný náhodný vektor a porovná jeho hodnotu s požadovanou hodnotou indexu spolehlivosti. Vzorek, který má nejmenší odchylku vypočteného indexu spolehlivost od definované hodnoty, je v případě splnění omezujících podmínek přijat jako nejlepší řešení v dané úrovni algoritmu AMS.

Program FNPO je sám tvořen pouze jediným spouštěcím souborem FNPO.exe. Aby FNPO fungoval správně, musí být umístěn ve stejné složce jako samotný program FReET (složka musí obsahovat potřebné soubory DLL a soubor Freet.exe, které jsou standardně umístěny ve složce, v níž je nainstalován FReET). Během svého fungování se FNPO odkazuje na soubory obsažené v aktuálním adresáři (tj. v adresáři z něhož je program spouštěn). Tímto adresářem by tedy měl být adresář shodný s místem instalace FReETu. V tomto adresáři musí být nakopírovaný také zdrojové soubory 1.fre a 2.fre. Výstupy programu FNPO jsou také automaticky ukládány do aktuálního adresáře.

2. DEFINICE ZDROJOVÝCH SOUBORŮ V PROGRAMU FREET

Před započetím práce s programem FNPO je nutné nadefinovat zdrojové soubory 1.fre a 2.fre. Pro správnou funkčnost programu je nutné dodržet uvedené názvy zdrojových souborů. Po nadefinování zmíněných souborů je nutné je nakopírovat do adresáře, v němž jsou uloženy programy FReET a FNPO.

Hlavní zdrojový soubor 1.fre (definice cílové funkce):

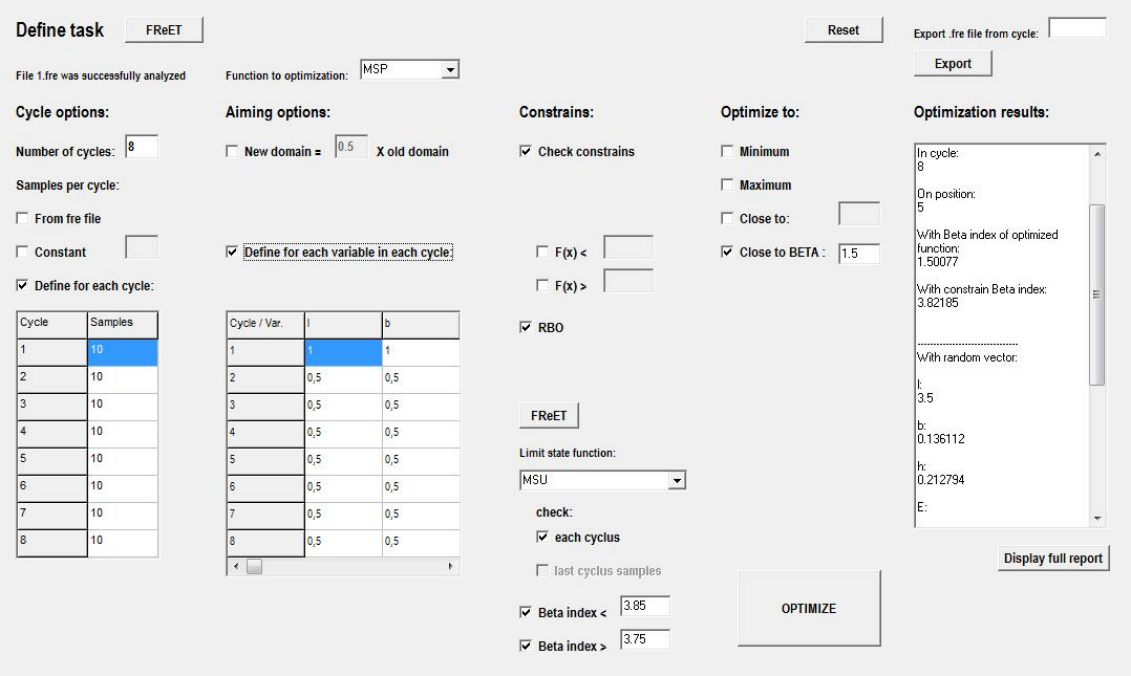
Je základním souborem potřebným vždy pro správnou funkci programu. Obsahuje definici cílové funkce a proměnných pro potřeby optimalizačního cyklu. Jako hlavní zdrojový soubor může sloužit jakýkoli .fre soubor, v němž nejsou při definici deterministických proměnných použity parametry. V případě využití možnosti optimalizovat samotnou hodnotu indexu spolehlivosti musí být v hlavním zdrojovém souboru definována příslušná funkce mezního stavu.

Vedlejší zdrojový soubor 2.fre (definice spolehlivostní omezující podmínky):

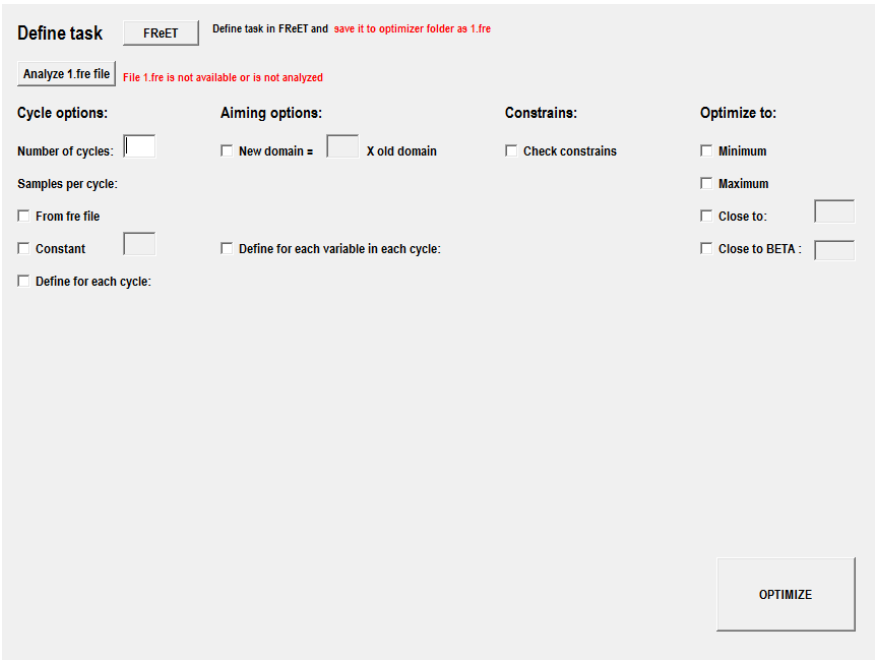
Je potřebný v případě využití spolehlivostní omezující podmínky. Tento soubor musí obsahovat definice všech proměnných s příslušnými statistickými parametry definujícími spolehlivostní úlohu. Musí také obsahovat funkci mezního stavu, pro niž se má počítat hodnota indexu spolehlivosti definující spolehlivostní omezení. Hlavní a vedlejší zdrojové soubory mohou být stejné nebo i velice odlišné v závislosti na definici úlohy spolehlivostní optimalizace.

3. UŽIVATELSKÉ ROZHRANÍ FNPO A ZPŮSOB OBSLUHY

Vzhled celého uživatelského rozhraní programu FNPO je patrný z Obr. 1.



Obr. 1 vzhled programu FNPO po zpřístupnění většiny voleb

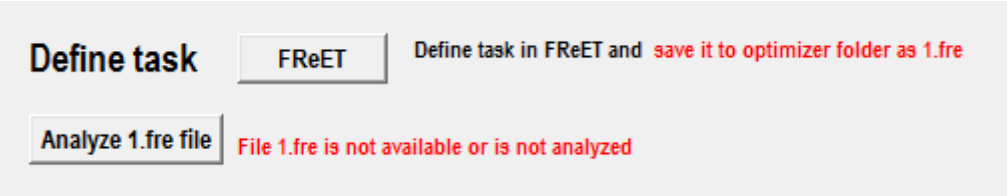


Obr. 2 vzhled programu FNPO po spuštění

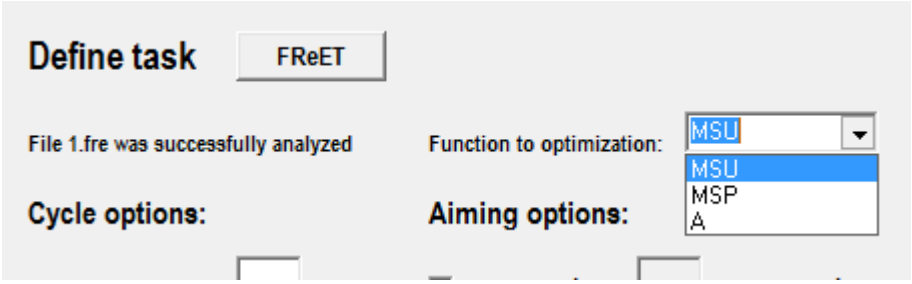
Program je koncipován tak, aby uživatel nemohl zadat protichůdné varianty nastavení. Jednotlivé volby v každé ze sekcí uživatelského prostředí jsou zpřístupňovány postupně,

v závislosti na předchozím nastavení jiných sekcí. Program například neumožní, aby bylo nastaveno jako cíl optimalizace současně minimum i maximum funkce atd. Celé rozhraní je koncipováno tak, aby se co nejvíce předešlo kolapsu programu vlivem chybného zadání vstupních dat. Po svém spuštění vypadá FNPO jako na Obr.2.

Program uživatele vybídne k definici hlavního zdrojového souboru a jeho uložení do aktuálního adresáře.



Stiskem tlačítka "FReET" se otevře standardní rozhraní programu FReET a uživatel může definovat soubor 1.fre. Po stisknutí tlačítka "Analyze 1.fre file" program načte potřebná data ze zdrojového souboru 1.fre. V případě neúspěchu akce program zobrazí chybové hlášení.

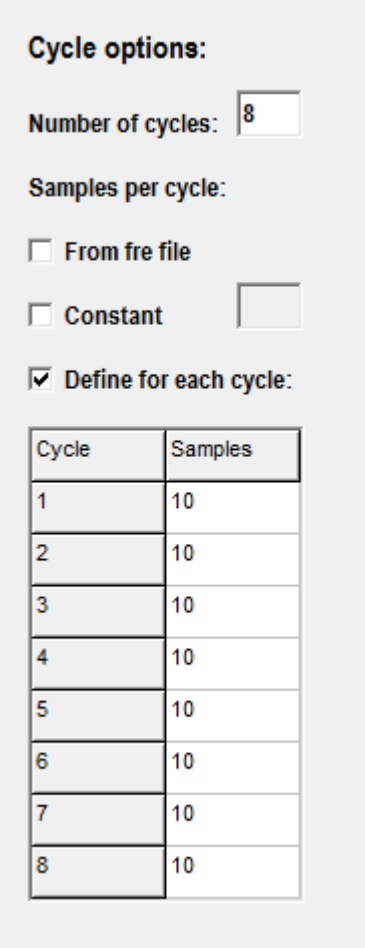


Po načtení souboru 1.fre se objeví zpráva, že analýza souboru proběhla úspěšně, zobrazí se výzva k selekci cílové funkce z množiny funkcí definovaných v souboru 1.fre. V pravém horním rohu uživatelského rozhraní také přibude tlačítko Reset, které vymaže data získaná analýzou souboru 1.fre a připraví program zpět do defaultního nastavení pro opětovnou analýzu souboru. Po jeho stisknutí se zobrazí dotaz, zda si přejeme vymazat stávající soubor 1.fre z aktuálního adresáře. V případě volby ne program projde soubor 1.fre a vymaže z něj výsledky z případného předchozího spuštění procesu optimalizace. Tím připraví soubor pro opětovné použití.

Následuje nastavení jednotlivých parametrů algoritmu AMS. Nejprve je třeba nastavit počet úrovní (cyklů) algoritmu AMS v kolonce "Number of cycles" v sekci "Cycle options". Program umožní uživateli zadat libovolné celé číslo. Následuje volba počtu vzorků generovaných v každém z cyklů algoritmu AMS. Jsou k dispozici tři možnosti:

- Zvolit počet vzorků definovaný ve hlavním zdrojovém souboru 1.fre – volba "From fre file".
- Zvolit počet vzorků jako konstantní pro každý z cyklů – volba "Constant".

- Zvolit počet vzorků zvlášť pro každý jednotlivý cyklus – volba “Define for each cycle“ (po jejím zatržení se zobrazí tabulka, v níž můžeme zadat požadovaný počet simulací pro každý z cyklů).



Cycle	Samples
1	10
2	10
3	10
4	10
5	10
6	10
7	10
8	10

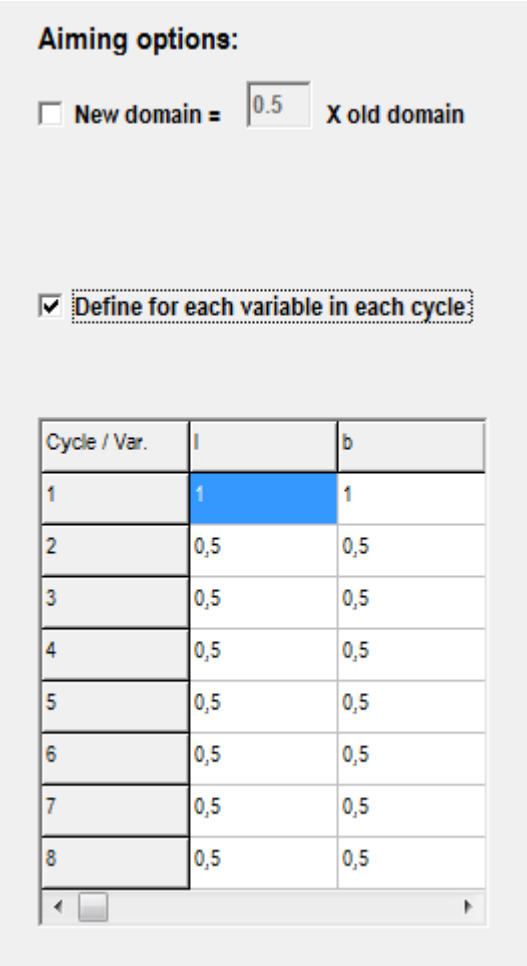
Program bere údaje o počtu simulací v každém z cyklů vždy z tabulky, která se zpřístupní po zaškrtnutí volby “Define for each cycle“. Při zaškrtnutí zbylých dvou voleb se tabulka vyplní automaticky. To umožňuje rychlé vyplnění tabulky v případě, kdy chce uživatel odlišit hodnotu počtu prováděných simulací pouze u několika úrovní. V takovém případě se nejdříve zaškrtne volba “Constant“ a zadá se požadovaný počet simulací, a poté se zaškrtne volba “Define for each cycle“ a změní se počty simulací jen u uživatelem definovaných cyklů.

V dalším kroku je nutné definovat postup redukce velikosti návrhového prostoru na jednotlivých úrovních algoritmu AMS. Jedná se tedy o specifikaci hodnoty $q \in (0, 1)$, kterou bude postupně násobena směrodatná odchylka každé z dimenzí návrhového prostoru při průběhu metody AMS. Jsou k dispozici dvě volby nastavení:

- Zvolit hodnotu q konstantní pro každou z dimenzí ve všech cyklech – volba “New domain = … x old domain“.

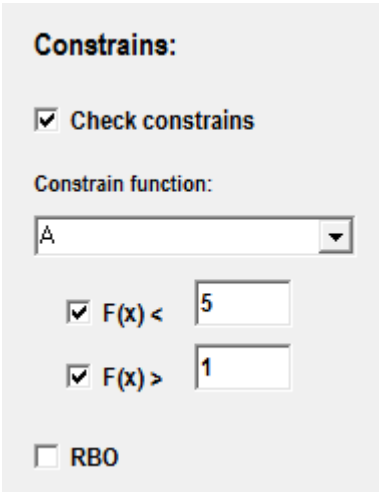
- Definovat hodnotu q pro každou proměnnou v každém z cyklů individuálně – volba “Define for each variable in each cycle“ (po jejím zaškrtnutí se zpřístupní tabulka, v níž můžeme příslušné hodnoty q specifikovat)

Stejně jako v případě nastavení sekce “Cycle options“ se i u “Aiming options“ bere hodnota q vždy z tabulky zpřístupněné po zatržení druhé z možných voleb. Zaškrtnutím první volby se tabulka opět vyplní automaticky.

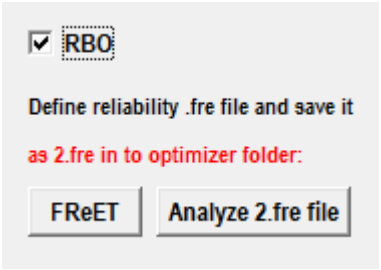


Další nastavitelnou sekcí je “Constrains“ tedy sekce, v níž je možné specifikovat omezující podmínku. Program FNPO ve své první verzi umožňuje zatím definovat pouze jednu omezující podmínku, a to buď ve formě omezení funkční hodnoty vybrané funkce definované ve hlavním zdrojovém souboru 1.fre, nebo omezením povoleného intervalu hodnot indexu spolehlivosti pro vybranou funkci mezního stavu definovanou ve vedlejším zdrojovém souboru 2.fre.

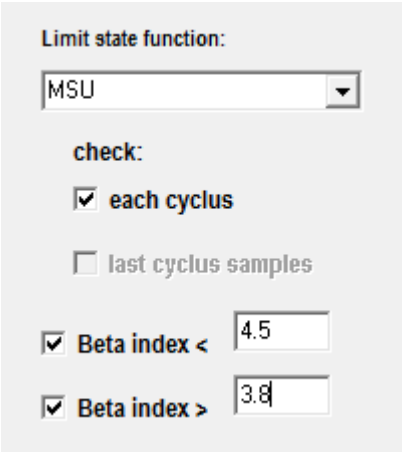
Zadání omezujících podmínek není povinné. Zadávací rozhraní se zpřístupní až po zatržení políčka “Check constarins“ v sekci “Constrains“.



Zaškrtnutím volby RBO (Relliability Based Optimization) se znepřístupní selekce omezující funkce pod políčkem "Check constrains" a zpřístupní se nová skupina voleb pro nastavení spolehlivostní omezující podmínky.

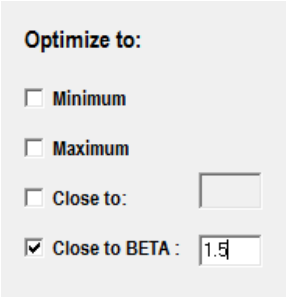


Program uživatele nejprve vyzve k definici vedlejšího zdrojového souboru 2.fre a jeho uložení do aktuálního adresáře. Tlačítko "FReET" vyvolá klasické uživatelské rozhraní programu FReET pro definici souboru 2.fre. Po stisknutí tlačítka "Analyze 2.fre file" program načte potřebná data ze souboru 2.fre. Následně se zpřístupní selekce funkce mezního stavu k výpočtu hodnoty indexu spolehlivosti sloužícího jako omezující podmínka.

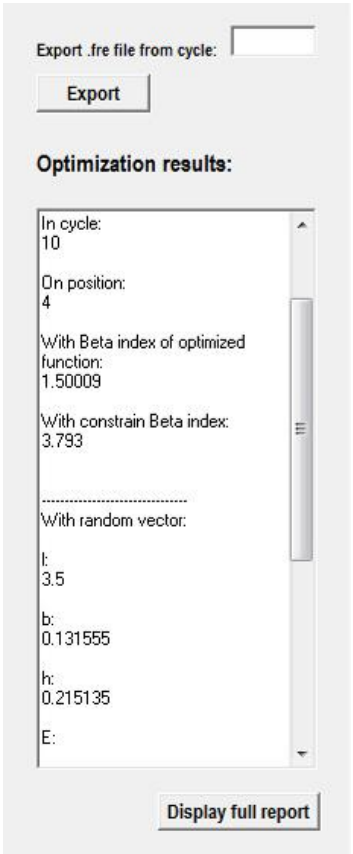


Poslední sekcí nastavení je “Optimize to“, která slouží k definici cíle optimalizace. Tato sekce je složena ze čtyř možných voleb:

- Cílem optimalizace je nalezení minimální hodnoty cílové funkce – volba “Minimum“.
- Cílem optimalizace je nalezení maximální hodnoty cílové funkce – volba “Maximum“.
- Cílem optimalizace je nalezení vektoru vstupních hodnot, který odpovídá dané funkční hodnotě cílové funkce – volba “Close to“.
- Cílem optimalizace je nalezení vektoru vstupních hodnot, který odpovídá dané hodnotě indexu spolehlivosti cílové funkce mezního stavu – volba “Close to BETA“.



Po nastavení všech potřebných parametrů je možné spustit proces optimalizace pomocí tlačítka “Optimize“ v pravém dolním rohu uživatelského rozhraní. Program provede optimalizaci dle uživatelem specifikovaných požadavků a zobrazí získané výsledky v textovém poli na pravé straně uživatelského rozhraní v sekci “Optimization results“.



Tlačítkem “Display full report“ uživatel otevře textový soubor “Optimization report.txt“ obsahující kompletní záznam všech .fre souborů generovaných v optimalizačním cyklu v rámci algoritmu AMS včetně všech příslušných výsledků. V případě optimalizace samotné hodnoty indexu spolehlivosti obsahuje tato zpráva i hodnoty indexů spolehlivosti cílové funkce mezního stavu vypočtené pro každou z generovaných realizací náhodného vektoru. Záznam je tříděn podle jednotlivých úrovní algoritmu AMS. Tlačítko “Export“ umožňuje extrakci .fre souboru z vybraného cyklu algoritmu AMS. Po jeho stisknutí se extrahovaný soubor automaticky otevře pro zpracování ve FReETu.

4. VÝSTUPNÍ SOUBORY

Zpráva “**Optimization report.txt**“ je hlavním výstupem optimalizačního procesu a je automaticky uložena v aktuálním adresáři. Kromě této zprávy program vytvoří také další soubory obsahující data potřebná ke zpětnému hodnocení procesu spolehlivostní optimalizace. Jsou to tyto textové soubory:

RBO list report.txt – soubor obsahující výpis všech .fre souborů generovaných pro výpočet indexů spolehlivosti k porovnání se spolehlivostní omezující podmínkou. Tento soubor je vytvořen pouze pokud je aktivní volba “RBO“ v sekci “Constrains“.

RBO result report.txt – soubor obsahující vypočtené hodnoty indexů spolehlivosti pro každou z vygenerovaných realizací náhodného vektoru v každém z cyklů algoritmu AMS. Tyto hodnoty slouží pro konfrontaci se spolehlivostní omezující podmínkou. Jedná se tedy o obecně jiné hodnoty indexů spolehlivosti než v souboru “Optimization report.txt“. Zpráva rovněž obsahuje informaci o tom, zda daná realizace vyhověla omezující podmínce či nikoliv.

Kromě těchto souborů program ukládá také dvě zprávy zálohující původní zdrojové soubory 1.fre a 2.fre. Těmito soubory jsou:

Analysis report.txt – zpráva obsahující původní soubor 1.fre a informace o jeho základních parametrech (počet a jména proměnných a funkcí, počet řádků atd…). Tento soubor je vytvořen po stisku tlačítka “Analyze fre file“ při prvním načtení hlavního zdrojového souboru 1.fre.

Analysis 2.fre report.txt – zpráva obsahující původní soubor 2.fre a informace o jeho základních parametrech. Je vytvořen po stisku tlačítka “Analyze 2.fre file“ v sekci “Constrains“.

Díky těmto souborům je možné celý proces spolehlivostní optimalizace podrobně sledovat v každém z jeho kroků. Výstupní soubory se přepíší při každé nové optimalizaci. V případě potřeby uchování výstupních souborů je nutné je přemístit z aktuálního adresáře programu FNPO před začátkem práce na jiném projektu.