Obsah

Značení

V této bakalářské práci je použito následující značení:

t	diskrétní časový okamžik
a_t	hodnota veličiny v čase t
$\mathop{\mathrm{E}}_{a}$	operátor střední hodnoty s rozdělením pravděpodobnosti P^a
$t\!:\!s$	posloupnost časů $(t, t+1, \ldots, s)$
$a_{t:s}$	posloupnost veličin $(a_t, a_{t+1}, \ldots, a_s)$
$g_{t:s}(a_{t:s})$	posloupnost funkčních hodnot $(g_t(a_t), g_{t+1}(a_{t+1}), \ldots, g_s(a_s))$
$\lvert H \rvert$	počet prvků v množině H

Úvod

V technické praxi, stejně jako běžném životě, jsme nuceni dělat rozhodnutí. Ať už se jedná o řízení výrobní linky či hledání optimálního spojení mezi dvěma místy, naše rozhodnutí vycházejí ze znalostí, které o světě máme. Chceme-li činit úspěšná rozhodnutí, je třeba vyřešit dvě úlohy: 1) řízený objekt co nejlépe poznat a 2) dosáhnout cíle, který jsme si vytyčili. Tyto dva úkoly jsou však většinou v rozporu: systém se nejlépe pozná, když se nechová podle našich požadavků. V reálném světě navíc existují náhodné jevy, poruchy a nepředvídané situace, které jednotně nazýváme neurčitostí. Tato skutečnost způsobuje, že naše znalost systému není nikdy dokonalá.

Za účelem řízení systémů, které jsou buď natolik složité, že jejich deterministický popis je nemožný, nebo obsahují náhodné prvky již ze své podstaty, vzniklo stochastické řízení, nebo-li optimální řízení za neurčitosti. Cílem stochastického řízení je minimalizovat velikost odchylek systému od požadovaného stavu optimalizací řídících zásahů.

Jeden z přístupů k řešení tohoto probému je dynamické programování, které navrhl americký matematik Richard Bellman[]. Jedná se o metodu, která s využitím zpětného chodu minimalizuje hodnotu očekávané ztátové funkce.

Tento přístup má analytické řešení pouze v případě znalosti všech parametrů systému, což je většinou nemožné. V šedesátých letech 20. století navrhl Alexander Aronovich Feldbaum řešení použitím takzvaného duálního řízení. Hlavní myšlenkou tohoto přístupu bylo, že řízení musí nejen minimalizovat aktuální ztrátu, ale rovněž musí získat o systému co nejvíce informací pro minimalizaci budoucích ztrát.

Přímá aplikace tohoto postupu je však bohužel i u poměrně jednoduchých značně komplikována složitostí výpočtu. K řešení úlohy je proto vhodné požít aproximačních metod.

Tato bakalářská práce si klade následující cíle

- Formulace úlohy stochastického řízení
- Řešení úlohy stochastického řízení s aditivní ztrátouvou funkcí pomocí duálního řízení
- Formulace úlohy stochastického řízení za neúplných informací a její převedení na úlohu s úplnými znalostmi systému

- Představení některých aproximačních přístupů k duálnímu řízení, zejména pak stochastického iterativního dynamického programování
- Aplixace duálního řízení k nalezení optimální strategie na jednoduchém systému
- Porovnání uvedených aproximačních přístupů na jednoduchém systému

Kapitola 1

Úloha stochastického řízení

DEFINICNI OBORY

1.1 Formulace úlohy stochastického řízení

Ústředním pojmem v teorii řízení je *systém*. Systém je část světa, kterou chceme poznat či řídit. Informace o stavu sytému získáváme prostřednictvím *výstupů*. V této kapitole budeme předpokládat, že můžeme stav systému měřit přímo. Případem nepřímého měření s neznámými parametry se zabývá následující kapitola. Řízení, tj. ovlivňování stavu systému, můžeme provádět pomocí *vstupů*.

Budeme-li předpokládat diskrétní povahu času, můžeme stav systém v časovéme okamžiku t podél řídícího horizontu délky N popsat systémem rovnic

$$x_{t+1} = f_k(x_t, u_t, w_t), \qquad t = 0, 1, \ldots, N-1, \tag{1.1}$$

kde x_t je stav systému v čase t, u_t je vstup v čase t a w_t náhodná veličina reprezentující přítomnost šumu.

Dále máme předepsanou ztrátovou funkci

$$g(x_{0:N}, u_{0:N-1}, w_{0:N-1}). \tag{1.2}$$

Posloupností řídících strategií $\pi = \mu_{0:N-1}$ budeme rozumět posloupnost zobrazení

$$\mu_t(x_t) = u_t \qquad t = 0, 1, \ldots, N-1, \tag{1.3}$$

PRIPUSTNE STRATEGIE

Pro danou řídící strategii označme očekávanou ztrátu jako

$$J_\pi(x_0) = \mathop{\mathrm{E}}_{w_{0:N-1}} \{g(x_{0:N}, \mu_{0:N-1}(x_{0:N-1}), w_{0:N-1})\} \tag{1.4}$$

Úlohou je potom najít takovou π^*, pro kterou platí

$$J_{\pi^*}(x_0) = \min_{\pi \in \Pi} J_\pi(x_0) \tag{1.5}$$

Celkově se tedy jedná o optimalizační úlohu nalézt takovou posloupnost funkcí (1.3), která minimalizuje očekávanou ztrátovu (1.4) za podmínek (1.1).

1.2 Použití dynamického programování při řešení úlohy stochastického řízení s aditivní ztrátou

Úlohu stochastického řízení tak, jak byla definována v předchozí části, nelze obecně řešit. Je tedy potřeba úlohu nějak blíže specifikovat. V tomto směru je možné omezit se na nějaký speciální tvar ztrátové funkce (1.4). Jako vhodné řešení se ukazuje uvažovat tzv. aditivní tvar ztrátové funkce, tedy že existují funkce g_t takové, že můžeme psát

$$g(x_{0:N}, u_{0:N-1}, w_{0:N-1}) = g_N(x_N) + \sum_{t=0}^{N-1} g_t(x_t, u_t, w_t) \tag{1.6}$$

Očekávanou ztrátu (1.4) potom můžeme přepsat do tvaru

$$J_\pi(x_0) = \underset{w_{0:N-1}}{\mathrm{E}} \left\{ g_N(x_N) + \sum_{t=0}^{N-1} g_t(x_t, \mu_t(x_t), w_t) \right\} \tag{1.7}$$

Takto specifikovaná úloha se dá řešit použitím dynamického programování []. Dynamické programování je přístup k řešení optimalizačních úloh, na které se můžeme dívat jako na posloupnost rozhodnutí, pro které platí tzv. princip optimality. Ten říká, že optimální posloupnost rozhodnutí má tu vlastnost, že pro libovolný počáteční stav a rozhudnutí musí být všechna následující rozhodnutí optimální vzhledem k výsledkům rozhodnutí prvního. Důkaz, že pro ztrátu tvaru (1.6) platí princip optimality je snadný a lze ho nalézt například v [].

Při řešení úlohy stochastického řízení s aditivní ztrátou je tedy možné postupovat, jak je u úloh řešených pomocí dynamického programování zvykem. Minimální hodnotu střední ztráty od okamžiku t do N v závislosti na x_t označíme $J_t(x_t)$. Můžeme pro ni psát

$$J_N(x_N) = g_N(x_N) \tag{1.8}$$

$$J_t(x_t) = \min_{u_t \in U(x_t)} \underset{w_t}{\mathrm{E}} \left\{ g_k(x_t, u_t, w_t) + J_{t+1}(f_t(x_t, u_t, w_t)) \right\} \qquad t = 0, \ldots, N-1 \tag{1.9}$$

Pří řešení budeme postupovat od konce řídícího horizontu a postupně hledat $J_t(x_t)$. Potom libovolná $\pi = \{\mu_0, \ldots, \mu_{N-1}\}$, která splňuje systém rovnic

$$J_t(x_t) = \underset{w_t}{\mathrm{E}} \left\{ g_k(x_t, \mu_t(x_t), w_t) + J_{t+1}(f_t(x_t, \mu_t(x_t), w_t)) \right\} \qquad t = 0, \ldots, N-1 \tag{1.10}$$

je optimální posloupnost rozhodnutí. Na systém rovnic (1.10) se tedy můžeme dívat jako na implicitní předpis pro π.

Kapitola 2

Úloha stochastického řízení s nepřesnými daty

Při aplikaci matematického modelování na řešení nějaké konkrétní úlohy se obvykle potýkáme s problémem, jak určit konstanty, které daný model určují. Zkoumáme-li například nějaký fyzikální systém, z rozboru fyzikálních zákonitostí obvykle známe tvar rovnic, které určují jeho vývoj v čase, nicméně počáteční podmínky či parametry, které v rovnicích vystupují a jsou pro daný systém charakteristické, můžeme získat pouze nepřímo, obvykle měřením vhodných veličin. Modifikací úlohy stochastického řízení pro případ přítomnosti neznámých parametrů se zabývá tato kapitola.

2.1 Formulace úlohy stochastického řízení s nepřesnými daty

Informace o stavu systému x_t v čase t získáváme pomocí výstupu y_t, který je dán jako

$$y_0 = h_0(x_0, v_0), \qquad y_t = h_t(x_t, u_{t-1}, v_t), \qquad t = 1, \dots, N-1, \tag{2.1}$$

kde v_t je náhodná veličina charakterizující chybu měření. Počáteční stav x_0 je dán rozdělením pravděpodobnosti P^{x_0} a další vývoj systému určuje soustava (1.1).

Informace, které jsou v průběhu řízení k dispozici je zvykem psát ve formě tzv. *informačního vektoru*, který má tvar

$$I_0 = y_0, \qquad I_t = (y_{0:t}, u_{0:t-1}), \qquad t = 1, \dots, N-1. \tag{2.2}$$

Řídící strategie $\pi = \mu_{0:N-1}$ nyní nemůže explicitně záviset na stavu systému, protože máme k dispozici pouze informační vektor. Hledáme tedy

$$\mu_t(I_t) = u_t \qquad t = 0, 1, \dots, N-1, \tag{2.3}$$

PRIPUSTNE STRATEGIE

Úkolem je najít přípustnou strategii (2.3), která by minimalizovala očekávanou ztrátu

$$J_\pi = \mathop{\mathrm{E}}_{\substack{x_0,\, w_{0:N-1},\\ v_{0:N-1}}} \left\{ g_N(x_N) + \sum_{t=0}^{N-1} g_t(x_t, \mu_t(I_t), w_t) \right\}, \tag{2.4}$$

za podmínek (1.1) a (2.1).

2.2 Převod na úlohu s úplnými daty

Protože v čase t nemáme k dispozici přímo stav systému x_t, ale pouze informační vektor I_t, nemůžeme použít postup z předchozí kapitoly. Před tím je potřeba úlohu vhodně transformovat. Za tímto účelem zapíšeme informační vektor ve tvaru

$$I_0 = y_0, \qquad I_{t+1} = (I_t, u_t, y_{t+1}), \qquad t = 1, \ldots, N-1. \tag{2.5}$$

Na tuto rovnost můžeme pohlížet jako na rovnice systému (1.1). Stav v čase t je nyní I_t, vstup u_t a y_{t+1} náhodná veličina podmíněná I_t a u_t přes (2.1).

Dále přejdeme k nové ztrátové funkci, kterou definujeme jako

$$\tilde{g}_N(I_N) = \mathop{\mathrm{E}}_{x_N} \{ g_N(x_N) | I_N \}, \tag{2.6}$$

$$\tilde{g}_t(I_t, u_t, w_t) = \mathop{\mathrm{E}}_{x_t} \{ g_t(x_t, u_t, w_t) | I_t, u_t \}, \qquad t = 1, \ldots, N-1. \tag{2.7}$$

Očekávanou ztrátu nyní můžeme psát ve tvaru

$$J_N(I_N) = \tilde{g}_N(I_N) \tag{2.8}$$

$$J_t(I_t) = \min_{u_t \in U_t} \mathop{\mathrm{E}}_{w_t, y_{t+1}} \{ \tilde{g}_t(I_t, u_t, w_t) + J_{t+1}((I_t, u_t, y_{t+1})) | I_t, u_t \} \qquad t = 0, \ldots, N-1 \tag{2.9}$$

Tato úloha již může být řešena pomocí dynamického programování. Pří řešení budeme postupovat od konce řídícího horizontu a postupně hledat $J_t(I_t)$. Potom libovolná $\pi = \{\mu_0, \ldots, \mu_{N-1}\}$, která nabývá minimální očekávané ztráty $J_0(y_0)$ je optimální posloupnost rozhodnutí.

2.3 Řízení systému s neznámými parametry

Pokud rovnice systému obsahuje nějaký neznámý parametr θ, můžeme využít znalosti řešení problému s neúplnými informacemi.

Hledané řízení by mělo nejen minimalizovat aktuální ztrátu, ale rovněž získat o systému co nejvíce informací pro minimalizaci budoucích ztrát. Tento postup se nazývá duální řízení [ref].

V úloze duálního řízení máme výstupy systému y_t popsány jako

$$y_0 = h_0(\theta, v_0), \qquad y_t = h_t(I_{t-1}, \theta, u_{t-1}, v_t), \qquad t = 1, \ldots, N-1, \tag{2.10}$$

Ztrátová funkce je nyní

$$g(y_{0:N}, u_{0:N-1}, v_{0:N-1}) = g_N(y_N) + \sum_{t=0}^{N-1} g_t(y_t, u_t, v_t). \tag{2.11}$$

Předpokládejme dále, že o parametru θ máme nějakou apriorní informaci θ_0 a odhadovací proceduru tvaru

$$\theta_{t+1} = f_t(I_t, \theta_t, u_t, y_{t+1}), \qquad t = 1, \ldots, N-1. \tag{2.12}$$

Rovnici (2.12) můžeme podobně jako (2.5) považovat za rovnici systému (1.1) pro stav (I_t, θ_t), vstup u_t s šumem y_{t+1}. Do rovnice (2.10) dosadíme za θ jeho aktuální odhad, tedy

$$y_0 = h_0(\theta_0, v_0), \qquad y_t = h_t(\theta_{t-1}, I_{t-1}, u_{t-1}, v_t), \qquad t = 1, \ldots, N-1, \tag{2.13}$$

Rovnice (2.12), (2.13) a (2.11) představují úlohu stochastického řízení s nepřesnými daty.

2.3.1 Bayesovské učení

Přímočarý postup, jak pro neznámý parametr θ získat aposteriorní hustotu pravděpodobnosti $f(\theta_{t+1}|I_t)$, je-li k dispozici apriorní hustota pravděpodobnosti $f(\theta_t)$ a informační vektor I_t, je aplikace Bayesova vzorce

$$f(\theta_{t+1}|I_t) = \frac{f(I_t|\theta_{t+1}) f(\theta_t)}{\int f(I_t|\theta_{t+1}) f(\theta_t) \mathrm{d}\theta_t} \tag{2.14}$$

Rekurzivní použití vzorce (2.14) pro odhad parametru θ je postup Bayesovského učení [ref].

Při konkrétním vypočtu má však tento přístup dvě nevýhody: 1) nikdy nemáme k dispozici $f(I_t|\theta_{t+1})$, ale pouze aproximaci z měření I_t a 2) aposteriorní hustota pravděpodobnosti nemusí mít analytické vyjádření, což její použití v dalším výpočtu komplikuje.

2.3.2 Kalmanův filtr

Pokud v rovnicích (2.10) popisujících výstup systému vystupuje gausovký šum a neznámý parametr je separován jako lineání člen, situace se značně zjednoduší.

Dle předpokladu má výstup v čase t tvar

$$y_{t+1} = \tilde{h}_t(I_t, u_t) + A_t(I_t, u_t))\theta_t + v_{t+1},, \qquad t = 0, \ldots, N-1. \tag{2.15}$$

kde $\tilde{h}_t(I_t, u_t)$, resp. $A_t(I_t, u_t)$ je známá funkce, resp. matice závisící na informačním vektoru a aktuální vstupu. Dále předpokládáme gausovské rozložení šumu v_{t+1} se známým rozptylem

$$v_{t+1} \sim N(0, Q_{t+1}), \tag{2.16}$$

gausovské rozložení odhadu neznámého parametru θ_t a jejich nekorelovanost, tedy

$$\theta_t \sim N(\hat{\theta}, P_t), \tag{2.17}$$
$$\mathrm{Cov}(v_{t+1}, \theta) = 0. \tag{2.18}$$

Budeme požadovat, aby odhadovací procedura (2.12) střední hodnoty parametru θ_{t+1} byla tvaru lineární opravy střední hodnoty θ_t úměrné neurčitosti v systému. Tedy že

$$\hat{\theta}_{t+1} = \hat{\theta}_t + K_t(y_{t+1} - \tilde{h}_t(I_t, u_t) - A_t\hat{\theta}_t), \tag{2.19}$$

kde K_t je neznámá matice, kterou určíme z požadavku minimalizace výsledné matice rozptylu P_{t+1}. Pro ni jako funkci K_t můžeme psát

$$P_{t+1}(K_t) = \mathrm{E}[(\theta - \hat{\theta}_{t+1})(\theta - \hat{\theta}_{t+1})^T]. \tag{2.20}$$

Dosazením za $\hat{\theta}_{t+1}$ z (2.19) a za y_t ze (2.15) a úpravou dostaneme (pro libovolnou matici B budeme pro lepší čitelnost namísto BB^T psát zkráceně B^2)

$$\begin{aligned} P_{t+1}(K_t) &= \mathop{\mathrm{E}}_{\theta, v_t}\left\{(\theta - \hat{\theta}_t - K_t(y_{t+1} - \tilde{h}_t(I_t, u_t) - A_t\hat{\theta}_t))^2\right\} \\ &= \mathop{\mathrm{E}}_{\theta, v_t}\left\{((I - K_tA_t)(\theta - \hat{\theta}_t) - K_tv_t)^2\right\} \\ &= (I - K_tA_t)\,\mathrm{E}\left\{(\theta - \hat{\theta}_t)^2\right\}(I - K_tA_t)^T - (I - K_tA_t)\,\mathrm{Cov}(\theta, v_t)K_t^T - \\ &- K_t\,\mathrm{Cov}(\theta, v_t)(I - K_tA_t)^T + K_t\,\mathrm{E}\left\{v_t^2\right\}K_t^T. \end{aligned} \tag{2.21}$$

Použitím definice P_t, Q_t a předpokladu $\mathrm{Cov}(\theta, v_t) = 0$ máme

$$P_{t+1}(K_t) = (I - K_tA_t)P_t(I - K_tA_t)^T + K_tQ_tK_t^T. \tag{2.22}$$

Protože požadujeme minimální rozptyl odhadu $\hat{\theta}_{t+1}$, určíme K_t z rovnice

$$\frac{\partial\,\mathrm{tr}(P_t)}{\partial K_t} = 0. \tag{2.23}$$

K provedením derivace použijeme vzorce*ODVOZENI BUDE ASI AZ V DODATKU*

$$\frac{\partial\,\mathrm{tr}(MXN)}{\partial X} = M^TN^T, \tag{2.24}$$
$$\frac{\partial\,\mathrm{tr}(MXNX^TO)}{\partial X} = M^TO^TXN + OMXN, \tag{2.25}$$

kde M, N a O jsou konstantní matice.

Tím získáme lineární rovnici pro K_t tvaru

$$-P_t^TA_t - P_tA_t + K_tA_tP_tK_t + K_tA_t^TP_tK_t + 2Q_tK_t = 0, \tag{2.26}$$

která má řešení

$$K_t = P_tA_t(A_t^TP_tA_t + Q_t)^{-1} \tag{2.27}$$

Dosazením (2.27) do (2.22) po úpravě dostaneme

$$P_{t+1} = (I - K_t A_t) P_t \tag{2.28}$$

Celkově tedy od původního odhadu parametru $N(\hat{\theta}_t, P_t)$ k novému $N(\hat{\theta}_{t+1}, P_{t+1})$ přejdeme pomocí

$$K_t = P_t A_t (A_t^T P_t A_t + Q_t)^{-1} \tag{2.29}$$

$$\hat{\theta}_{t+1} = \hat{\theta}_t + K_t (x_{t+1} - f_t(x_t, u_t) - A_t \hat{\theta}_t) \tag{2.30}$$

$$P_{t+1} = (I - K_t A_t) P_t \tag{2.31}$$

Tato odhadovací procedura se nazývá Kalmanův filtr [ref].

Kapitola 3

Suboptimální přístupy k úloze duálního řízení

Ačkoliv použití dynamického programování přináší významný pokrok v řešení úlohy duálního řízení, analytické řešení obvykle není možné získat. V každém časovém kroku se totiž potýkáme se dvěma obecně obtížnými problémemy: 1) výpočet střední hodnoty a 2) minimalizace vzhledem k u_t. Oba problémy obecně nemají analytické řešení a bez další specifikace úlohy je proto třeba přejít k aproximačním metodám.

V této kapitole se předkládá popis několika možných přístupů k aproximativnímu řešení úlohy duálního řízení. Připomeňme, že úlohou duálního řízení je nalezení řídící strategie $\pi = \mu_{0:N-1}$, která by minimalizovala očekávanou ztrátu

$$J_\pi = \mathop{\mathrm{E}}_{\theta_0, v_{0:N-1}} \left\{ g_N(y_N) + \sum_{t=0}^{N-1} g_t(y_t, \mu_t(I_t, \theta_t), v_t) \right\}, \tag{3.1}$$

za apriorní informace θ_0 a podmínek

$$\theta_{t+1} = f_t(I_t, \theta_t, u_t, y_{t+1}), \tag{3.2}$$

$$y_0 = h_0(\theta_0, v_0), \qquad y_{t+1} = h_t(I_t, \theta_t, u_t, v_{t+1}), \qquad t = 0, \ldots, N-1. \tag{3.3}$$

$$v_{t+1} \sim N(0, Q_{t+1}) \tag{3.4}$$

$$\theta_t \sim N(\hat{\theta}, P_t), \tag{3.5}$$

$$\mathrm{Cov}(v_{t+1}, \theta) = 0. \tag{3.6}$$

Úlohu řešíme pomocí dynamického programování, tedy postupnou minimalizací očekávané ztráty od konce řídícího horizontu

$$J_N(I_N, \theta_N) = \mathop{\mathrm{E}}_{\theta_N, v_N} \{ g_N(y_N) \}, \tag{3.7}$$

$$J_t(I_t, \theta_t) = \min_{u_t \in U_t} \mathop{\mathrm{E}}_{y_{t+1}, v_t} \{ g_t(y_t, u_t, v_t) + J_{t+1}((I_t, , u_t, y_{t+1}, \theta_{t+1})) | I_t, \theta_t, u_t \}, \tag{3.8}$$

$$t = 0, \ldots, N-1, \tag{3.9}$$

kde θ_{t+1} a y_{t+1} se počítá dle (3.2) a (3.3).

3.1 Certainty equivalent control

Při použití metody Certainty equivalent control (CEC) [ref] se v rovnici pro očekávanou ztrátu nahradí náhodné veličiny svými středními hodnotami. Očekávaná ztráta tak přejde v

$$J_N(I_N, \theta_N) = g_N(y_N), \tag{3.10}$$

$$J_t(I_t, \theta_t) = \min_{u_t \in U_t} \left\{ g_t(y_t, u_t, \hat{v}_t) + J_{t+1}(I_t, \theta_{t+1}, u_t, \hat{y}_{t+1})) | I_t, \theta_t, u_t \right\}, \tag{3.11}$$

$$t = 0, \ldots, N-1, \tag{3.12}$$

3.2 Metoda separace

Při použití metody separace [ref] je proces řízení rozdělen do dvou fází: 1) indentifikace neznámého parametru a 2) řízení za použití odhadu $\hat{\theta}$ z první fáze.

První fáze slouží k nezávislému sběru dat, která jsou následně použita k odhadu neznámého parametru. K odhadu můžeme použít například rovnici (3.2). V druhé fázi pak po zbytek řídícího horizontu použijeme pro návrh řídící strategie odhad $\hat{\theta}$ z první fáze.

3.3 SIDP

Metoda stochastického iterativního dynamického programování (SIDP) [ref] spočívá v současném použití metody Monte Carlo k získání aproximace pro očekávanou ztrátu a iterativního dynamického programování k nalezení optimální strategie.

3.3.1 Metoda Monte Carlo

Metoda Monte Carlo je statistická simulační metoda, kterou navrhl ... [ref]. Její princip spočívá ve vzorkování nějaké náhodné veličiny za účelem odhadu její hledané charakteristiky, např. střední hodnoty.

V této práci je metoda Monte Carlo použita k výpočtu očekávané ztráty (3.1). Pří běžném použití dynamického programování máme při výpočtu $J_t(I_t, \theta_t)$ k dispozici předpis pro následující očekávanou ztrátu $J_{t+1}(I_{t+1}, \theta_{t+1})$. Metoda monte Carlo nám však dá k dispozici pouze odhad očekávané ztráty a použití těchto aproximací v dalším výpočtu by chybu výpočtu navyšovalo. Namísto toho se pro další výpočet uchovávají $\mu_t(I_t, \theta_t)$ a očekávaná ztráta v čase t se pak počítá jako průměr přes n realizací náhodné veličiny $(\theta_{t:N-1}, v_{t:N})$, tedy

$$\frac{1}{n} \sum_{i=1}^{n} \left(g_N(y_N^i) + \sum_{j=t}^{N-1} g_j(y_j^i, \mu_j(I_j^i, \theta_j), v_j^i) \right), \tag{3.13}$$

kde y^i_{j+1} se počítá podle (2.13) jako

$$y^i_{j+1} = h_j(I^i_j, \theta^i_j, \mu(I^i_j, \theta_j), v^i_{j+1}), \qquad j = t, \ldots, N-1, \qquad i = 1 \ldots, n, \tag{3.14}$$

a index i označuje i-tou realizaci dané veličiny. Realizace $\theta_{t:N-1}$ se generují podél trajektorie (3.3). To znamená, že dané θ_{k+1} se generuje až ve chvíli, kdy je známé I_k, u_k, rozdělení θ_k a y_{k+1} a tedy přes (3.2) i rozdělení θ_{k+1}.

Tento jednoduchý postup lze vylepšit víceúrovňovým porovnáním. Jedním z možných vylepšeních je dvouúrovňový algoritmus poposaným [ref]. V tomto algoritmu se nejprve pro každého kandidáta vygeneruje n_0 realizací. Na základě realizací se vyberou ti kandidáti, na který je nabyto minima s pravděpodobností větší než je daná mez α_0. Pro tyto se v druhé fázi vygeneruje dostatečný počet realizací tak, aby bylo možné nejlepší rozhodnutí zvolit s pravděpodobností alespoň rovné zadané mezi α_1. Takto upraveny algoritmus metody Monte Carlo je robustnější a umožňuje porovnání většího množství kandidátů, neboť počet realizací v první fázi může být poměrně nízký, slouží pouze k odfiltrování zjevně horších kandidátů na řízení. Pro účely této práce nicméně postačuje základní verze metody Monte Carlo a je proto v následující implementaci SIDP použita.

3.3.2 Iterativní dynamické programování

Iterativní dynamické programování [ref] je jední z přístupů k nalezení optimální strategie, která minimalizuje očekávanou ztrátu (3.1). Oproti dynamickému programování se problém řeší iterativně. Na začátku se zvolí nějaká apriorní strategie. V každé iteraci se potom vychází ze strategie spočtené v předchozím kroku a prostřednictvím perturbací tohoto (suboptimálního) řešení se hledá strategie, pro kterou bude očkávaná ztráta nižší. Tato se použije v následující iteraci.

3.3.3 Diskretizace prostoru

Při hledání optimální strategie $\mu_t(I_t, \theta_t)$ bychom pro přesné vyčíslení očekávané ztráty (3.13) na úseku řídícího horizontu $t : N$ potřebovali její analytické vyjádření. To ale není obvykle možné. Je proto nutné přejít k nějaké aproximaci, například 1) předpokládat nějaký tvar optimální strategie a při výpočtu určit pouze konstanty, které výslednou strategii určí jednoznačně, nebo 2) diskretizovat prostor (I_t, θ_t) a počítat $\mu_t(I_t, \theta_t)$ jen v bodech diskretizace a jinde se uchýlit k interpolaci (popřípadě extrapolaci). V této práci volíme druhou zmíněnou metodu. Poznamenejme, že díky předpokladu gaussovského rozdělení parametru θ_t, diskretizace vyhledem k θ_t znamená diskretizaci vzhledem k $(\hat{\theta}_t, P_t)$.

Jakým způsobem efektivně diskretizovat prostor nezávislých proměnných pro aproximativní výpočet očekávané ztráty (3.13) je při použití dynamického programování obtížná otázka. Bude-li bodů v diskretizaci příliš málo, bude výpočet nespolehlivý, naopak pro příliš jemnou diskretizaci bude časová náročnost výpočtu rychle stoupat (o časové náročnosti SIDP viz dále). Zde se ukazuje výhodnost použití iterativního dynamického programování, neboť stačí diskretizovat jen tu část prostoru která

bude potřebná v následující iteraci. Pomocí strategie spočtené v předchozím kroku a náhodných realizací šumu $v_{0:N}$ a neznámého parametru $\theta_{0:N}$ vygenerujeme trajektorie v $(I, \theta)_{0:N}$. V každé časové úrovni pak diskretizujeme jen tu část prostoru, která byla zasažena.

V této práci je volena jednoduchá metoda v které se spočte nejmenší hyperkvádr kolem zasažené tak, že se vezme nejmenší hyperkvádr orientovaný ve směru souřadných os, do kterého se vygenerované body vejdou. Prostor se poté diskretizuje pouze v této oblasti. Metodu k určení hyperkvádru s obecnou orientací lze najít v [ref].

3.3.4 Algoritmus SIDP

V tomto odílu je popsán algoritmus SIDP. Jeho parametry jsou

- n_{pass}, n_{iter}– počet opakování a iterací algoritmu
- N – řídící horizont
- n_g – počet bodů v diskretizaci každé dimenzi H_t, tj. $|H_t| = n_g^{\dim H_t}$
- $\pi^* = \mu_{0:N-1}(H_{0:N-1})$ – apriorní řídící strategie
- m – počet kadnidátů na změnu řídícího zásahu v jedné iteraci IDP
- β^{in} – počáteční rozsah pro hledání optimálního řídícího zásahu
- γ, λ – parametry pro redukci β^{in}
- n – počet realizací pro odhad metodou Monte Carlo

Jak plyne z následujícího popisu, časová složitost SIDP vzhledem k jeho parametrům je $O(n_{pass} n_{iter} N^2 m n_g^{\dim H_N})$ (časová náročnost metody Monte Carlo je úměrná vzdálenosti od konce horizontu).

for $i = 1$ to n_{pass} **do**
  for $j = 1$ to n_{iter} **do**
    $\beta_{i,j} := \gamma^{j-1}\lambda^{i-1}\beta^{in}$
    for $k = 1$ to $|H_t|$ **do**
      spočti trajektorii $H_{0,k}$, použij aktuální π^*, její interpolace a extrapolace a realizace neznámého parametru $\theta_0, \ldots, \theta_{N-1}$ podél této trajektorie
    end for
    for $t = N - 1$ to 0 **do**
      vytvoř $\tilde{H}_t$ jakožto rovnoměrnou síť v oblasti bodů H_t
      interpoluj (extrapoluj) $\mu_t^*(H_t)$ na $\mu_t^*(\tilde{H}_t)$
      for $k = 1$ to $|H_t|$ **do**
        for $m = -\left[\frac{m-1}{2}\right]$ to $\left[\frac{m}{2}\right]$ **do**
          pro $\tilde{H}_{t,k}$ vygeneruj kandidáta na řízení $\mu_t(\tilde{H}_{t,k}) = \mu_t^*(\tilde{H}_{t,k}) + m\beta_{i,j}$
          pomocí metody Monte Carlo spočti očekávanou ztrátu
        end for
        rozhodnutí s nejnižší očekávanou ztrátou uchovej jako nové optimální rozhodnutí pro $\tilde{H}_{t,k}$.
      end for
    end for
  end for
end for

Kapitola 4

Srovnání suboptimální přístupů při řízení jednoduchého systému

4.1 Popis systému

4.2 Transformace systému

4.3 Srovnání jednotlivých přístupů

Závěr

Sem prijde zaver

tady budou reference