Obsah

Značení

V této bakalářské práci je použito následující značení:

t	diskrétní časový okamžik
a_t	hodnota veličiny v čase t
$\underset{a}{\mathrm{E}}$	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 [3]. Jedná se o metodu, která s využitím zpětného chodu minimalizuje hodnotu očekávané ztátové funkce.

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

V šedesátých letech 20. století navrhl Alexander Aronovich Feldbaum řešení použitím takzvaného duálního řízení [5]. 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.

Tato bakalářská práce si klade následující cíle

- Formulace úlohy stochastického řízení
- Řešení úlohy stochastického řízení s aditivní ztrátouvou funkcí pomocí dynamického programování
- Formulace úlohy stochastického řízení s neúplným pozorováním a její převedení na úlohu s úplnými znalostmi systému
- Představení některých suboptimálních přístupů k úloze stochastického řízení

- Aplikace a porovnání zmíněných metod k nalezení optimální strategie 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. Budeme-li předpokládat diskrétní povahu času, stav systému v časovém okamžiku t podél řídícího horizontu délky N popisuje systém 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. V této kapitole budeme předpokládat, že můžeme stav systému pozorovat. Případem neúplného pozorování se zabývá následující kapitola.

V úloze řízení máme vždy předepsanou ztrátovou (resp. účelovou) funkci

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

Označme $U(x_t)$ množinu přípustných řídících zásahů pro systém ve stavu x_t. Přípustnou řídící strategií $\pi = \mu_{0:N-1}$ budeme rozumět posloupnost zobrazení

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

kde $u_t \in U(x_t)$ je přípustný řídící zásah.

Pro danou řídící strategii označme očekávanou ztrátu jako

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

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

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

kde Π značí množinu všech přípustných řídících strategií.

Celkově se tedy jedná o optimalizační úlohu nalézt takovou posloupnost funkcí (1.3), která minimalizuje očekávanou ztrátovou (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 se ukazuje uvažovat tzv. aditivní tvar ztrátové funkce, tedy že existují funkce g_t takové, že můžeme psát

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

Očekávanou ztrátu (1.4) potom můžeme přepsat do tvaru

$$J_\pi(x_0) = \mathop{\mathrm{E}}_{w_{0:N-1}} \left\{ \sum_{t=0}^{N-1} g_t(x_{t+1}, \mu_t(x_t)) \right\}. \tag{1.7}$$

Takto specifikovaná úloha se dá řešit použitím dynamického programování [3]. 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 [4].

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

$$J_N(x_N) = 0 \tag{1.8}$$

$$J_t(x_t) = \min_{u_t \in U(x_t)} \mathop{\mathrm{E}}_{w_t} \{ g_k(x_{t+1}, u_t) + J_{t+1}(x_{t+1}) \} \qquad t = 0, \ldots, N-1. \tag{1.9}$$

Prí řešení budeme postupovat od konce řídícího horizontu a postupně hledat $J_t(x_t)$. Pro výpočet x_{t+1} se použije rovnice (1.1). Libovolnou řídící strategii $\pi = \{\mu_0, \ldots, \mu_{N-1}\}$, která splňuje systém rovnic

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

pak nazveme optimální posloupností rozhodnutí.

Kapitola 2

Úloha stochastického řízení s neúplným pozorováním

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. Tato kapitola se zabývá modifikací úlohy stochastického řízení pro případ přítomnosti neznámých parametrů.

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

$$y_0 = h_0(x_0, v_0), \qquad y_{t+1} = h_{t+1}(x_{t+1}, u_t, v_{t+1}), \qquad t = 1, \dots, N-1, \tag{2.1}$$

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

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

Řídící zásah nyní nemůže explicitně záviset na stavu systému, protože máme k dispozici pouze informační vektor. Podobně jako v předešlé kapitole proto zavádíme množinu $U(I_t)$ všech přípustných řídících zásahů za informace I_t a přípustnou řídící strategií bude $\pi = \mu_{0:N-1}$

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

kde $u_t \in U(I_t)$ je přípustný řídící zásah.

Úkolem je najít přípustnou strategii, která by minimalizovala očekávanou ztrátu

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

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

$$\tilde{g}_t(I_{t+1}, u_t) = \mathop{\mathrm{E}}_{x_{t+1}} \left\{ g_t(x_{t+1}, u_t) | I_t, u_t \right\}, \qquad t = 1, \ldots, N-1, \tag{2.6}$$

kde x_{t+1} se počítá dle (1.1) a x_t se považuje za náhodnou veličinu podmíněnou informačním vektorem I_t.

Očekávanou ztrátu nyní můžeme psát ve tvaru

$$J_N(I_N) = 0 \tag{2.7}$$

$$J_t(I_t) = \min_{u_t \in U_t} \mathop{\mathrm{E}}_{w_t, y_{t+1}} \left\{ \tilde{g}_t(I_{t+1}, u_t) + J_{t+1}(I_{t+1}) | I_t, u_t \right\} \qquad t = 0, \ldots, N-1 \tag{2.8}$$

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 řídící strategie.

2.3 Řízení systému s neznámými parametry

Pokud chceme řídit systém, jehož výstup závisí na nějakém neznámém konstantím parametru θ, můžeme využít znalosti řešení problému s neúplným pozorováním. Parametr θ bude reprezentovat stav systému x_t, který se nyní v čase nemění.

V této úloze máme výstupy systému y_t popsány jako

$$y_0 = h_0(\theta, v_0), \qquad y_{t+1} = h_t(I_t^{(d)}, \theta, u_t, v_{t+1}), \qquad t = 0, \ldots, N-1, \tag{2.9}$$

kde $I_t^{(d)} = (y_{t:t-d}, u_{t-1:t-d})$ a číslo d se nazývá řád modelu.

Označme T_t dostatečnou statistiku pro parametr θ založenou na informacích dostupných v čase t. Pokud dostatečná statistika neexistuje, pak bude T_t označovat nějakou její vhodnou aproximaci. Označme dále $H_t = (I_t^{(d)}, T_t)$ tzv. hyperstav systému.

Předpokládejme dále, že o parametru θ máme nějakou apriorní informaci v podobě hustoty pravděpodobnosti $f(\theta|T_0)$. Aposteriorní hustotu $f(\theta|T_{t+1})$ získáme pomocí Bayesova vzorce

$$f(\theta|T_{t+1}) = \frac{f(y_{t+1}|\theta, I_t^{(d)}, u_t) f(\theta|T_t)}{\int f(y_{t+1}|\theta, I_t^{(d)}, u_t) f(\theta|T_t) \mathrm{d}\theta} \tag{2.10}$$

Rekurzivní použití vzorce (2.10) pro odhad parametru θ se nazývá postup Bayesovského učení [10].

Pro vývoj hyperstavu H_t v čase můžeme na základě (2.10) psát

$$H_{t+1} = f_t(H_t, u_t, y_{t+1}), \qquad t = 1, \ldots, N-1. \tag{2.11}$$

Rovnici (2.11) můžeme podobně jako (2.5) považovat za rovnici systému (1.1) pro stav H_t a vstup u_t s šumem y_{t+1}.

Ztrátová funkce je nyní

$$g(y_{1:N}, u_{0:N-1}) = \sum_{t=0}^{N-1} g_t(y_{t+1}, u_t). \tag{2.12}$$

Úlohou je nalezení řídící strategie $\pi = \mu_{0:N-1}$, která by minimalizovala očekávanou ztrátu

$$J_\pi = \mathop{\mathrm{E}}_{\theta_0, v_{0:N-1}} \left\{ \sum_{t=0}^{N-1} g_t(y_{t+1}, \mu_t(H_t)) \right\}, \tag{2.13}$$

za apriorní informace $f(\theta|T_0)$, známého rozdělení šumu v_t a podmínek (2.11) a (2.9).

Rovnice (2.11), (2.9) a (2.12) potom představují úlohu stochastického řízení s nepřesnými daty.

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

$$J_N(H_N) = 0 \tag{2.14}$$

$$J_t(H_t) = \min_{u_t \in U_t} \mathop{\mathrm{E}}_{y_{t+1}} \left\{ g_t(y_{t+1}, u_t) + J_{t+1}(H_{t+1}) | H_t, u_t \right\}, \qquad t = 0, \ldots, N-1, \tag{2.15}$$

kde H_{t+1} se počítá dle (2.11). Střední hodnota vzhledem k y_{t+1} se počítá pomocí (2.9) a $f(\theta|T_t)$ jakožto aktuálního odhadu na parametr θ.

2.3.1 Kalmanův filtr

Pokud v rovnicích (2.9) popisujících výstup systému vystupuje aditivní gaussovký šum a neznámý parametr je separován jako lineání člen, můžeme vypočítat konkrétní tvar rovnice (2.11), tzv. Kalmanův filtr [7].

Dle předpokladu má výstup v čase t tvar

$$y_{t+1} = \tilde{h}_t(I_t, u_t) + A_t(I_t, u_t)\theta + v_{t+1},, \qquad t = 0, \ldots, N-1. \tag{2.16}$$

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 gaussovské rozložení šumu v_{t+1} se známým rozptylem

$$v_{t+1} \sim N(0, Q_{t+1}), \tag{2.17}$$

gaussovské rozložení odhadu neznámého parametru θ_t a jejich nekorelovanost, tedy

$$\theta_t \sim N(\hat{\theta}_t, P_t), \tag{2.18}$$

$$\mathrm{Cov}(v_{t+1}, \theta_t) = 0. \tag{2.19}$$

Dosazením do (2.10) se odvodí, že aposteriorní hustota pravděpodobnosti $f(\theta|T_{t+1})$ je rovněž gaussovská a její parametry $(\hat{\theta}_{t+1}, P_{t+1})$ splňují rovnice

$$K_t = P_t A_t (A_t^T P_t A_t + Q_t)^{-1} \tag{2.20}$$

$$\hat{\theta}_{t+1} = \hat{\theta}_t + K_t(y_{t+1} - \tilde{h}_t(I_t, u_t) - A_t\hat{\theta}_t), \tag{2.21}$$

$$P_{t+1} = (I - K_t A_t) P_t. \tag{2.22}$$

Odvození lze nalézt v [10].

Alternativní odvození bez požadavku gaussovského šumu je možné provést za předpokladu, že odhadovací proceduru střední hodnoty $\hat{\theta}_{t+1}$ neznámého parametru θ budeme hledat ve tvaru lineární opravy střední hodnoty $\hat{\theta}_t$ úměrné neurčitosti v systému. Tedy že

$$\hat{\theta}_{t+1} = \hat{\theta}_t + K_t(y_{t+1} - \mathop{\mathrm{E}}_{\theta, v_t} y_{t+1}), \tag{2.23}$$

kde K_t je neznámá matice, kterou určíme z požadavku minimalizace výsledné matice rozptylu P_{t+1}. Pro šum v_t budeme požadovat nulovou střední hodnotu a existenci druhého momentu. Matici rozptylu označíme opět Q_t.

Pro matici P_{t+1} jako funkci K_t můžeme psát

$$P_{t+1}(K_t) = \mathrm{E}[(\theta - \hat{\theta}_{t+1})(\theta - \hat{\theta}_{t+1})^T]. \tag{2.24}$$

Dosazením za $\hat{\theta}_{t+1}$ z (2.23) a za y_t ze (2.16) a úpravou dostaneme (pro libovolnou matici B budeme pro lepší čitelnost namísto BB^T psát zkráceně B^2)

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

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

$$P_{t+1}(K_t) = (I - K_t A_t) P_t (I - K_t A_t)^T + K_t Q_t K_t^T. \tag{2.25}$$

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

$$\frac{\partial \operatorname{tr}(P_t)}{\partial K_t} = 0. \tag{2.26}$$

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

$$\frac{\partial \operatorname{tr}(MXN)}{\partial X} = M^T N^T, \tag{2.27}$$

$$\frac{\partial \operatorname{tr}(MXNX^T O)}{\partial X} = M^T O^T XN + OMXN, \tag{2.28}$$

kde M, N a O jsou konstantní matice.

Tím získáme lineární rovnici pro K_t tvaru

$$-P_t^T A_t - P_t A_t + K_t A_t P_t K_t + K_t A_t^T P_t K_t + 2Q_t K_t = 0, \tag{2.29}$$

která má řešení

$$K_t = P_t A_t (A_t^T P_t A_t + Q_t)^{-1} \tag{2.30}$$

Dosazením (2.30) do (2.25) po úpravě dostaneme

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

Rovnice (2.23), (2.30) a (2.31) představují rovnice Kalmanova filtru.

Kapitola 3

Suboptimální přístupy k úloze duálního řízení

Ačkoliv použití dynamického programování přináší významný pokrok v řešení úlohy stochastického 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. Přípomeňme, že úlohou duálního řízení je nalezení řídící strategie $\pi = \mu_{0:N-1}$, která by minimalizovala očekávanou ztrátu

$$J_\pi = \mathop{\mathrm{E}}_{\theta_0, v_{0:N-1}} \left\{ \sum_{t=0}^{N-1} g_t(y_{t+1}, \mu_t(H_t)) \right\}, \tag{3.1}$$

za apriorní informace θ_0 a podmínek

$$H_{t+1} = f_t(H_t, u_t, y_{t+1}), \tag{3.2}$$

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

kde $H_t = (I_t^{(d)}, T_t)$ je hyperstav systému a T_t dostatečná statistika pro neznámý parametr θ v čase t.

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

$$J_t(H_t) = \min_{u_t \in U_t} \mathop{\mathrm{E}}_{y_{t+1}} \left\{ g_t(y_{t+1}, u_t) + J_{t+1}(H_{t+1}) | H_t, u_t \right\}, \qquad t = 0, \ldots, N-1, \tag{3.4}$$

kde T_{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) se v rovnici pro očekávanou ztrátu nahradí náhodná veličina y_{t+1} střední hodnotou $\hat{y}_{t+1}$. Ta se vypočítá z (3.3) pomocí známých rozdělení na v_t a postačující statistiky T_t. Očekávaná ztráta (1.4) tak přejde v

$$J_t(H_t) = \min_{u_t \in U_t} \left\{ g_t(\hat{y}_t, u_t) + J_{t+1}(\hat{H}_{t+1}) | H_t, u_t \right\}, \qquad t = 0, \dots, N-1, \tag{3.5}$$

Podrobnější pojednání s diskuzí aspektů použití CEC lze nalézt v [4].

3.2 Metoda separace

Při použití metody separace 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 Duální řízení

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]. ODKAZ NA FILDEBAUMA, POPIS PRINCIPU... (napr JEDNOKROKOVA OPTIMALIZACE S BUZENIM - FILATOV)

3.4 Metoda Monte Carlo

Metoda Monte Carlo [6] je statistická simulační metoda. 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 (3.4).

Pří běžném použití dynamického programování máme při výpočtu $J_t(H_t)$ k dispozici předpis pro následující očekávanou ztrátu $J_{t+1}(H_{t+1})$. 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 $\mu_t(H_t)$ a očekávaná ztráta v čase t se pak počítá jako průměr přes n realizací náhodných veličiny přes které je prováděna střední hodnota $(\theta_{t:N-1}, v_{t:N})$, tedy

$$\frac{1}{n} \sum_{i=1}^{n} \left(\sum_{j=t}^{N-1} g_j(y_{j+1}^i, \mu_j(H_j^i) \right), \tag{3.6}$$

kde y_{j+1}^i se počítá podle (3.3) jako

$$y_{j+1}^i = h_j(I_j^i, \theta_j^i, \mu(H_j^i), v_{j+1}^i), \qquad j = t, \dots, N-1, \qquad i = 1 \dots, n, \tag{3.7}$$

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, postačující statistika T_k a y_{k+1} a tedy přes (3.2) i hustota pravděpodobnosti $f(\theta_{k+1})$.

Tento jednoduchý postup lze vylepšit víceúrovňovým porovnáním kandidátů na optimální řízení. Jedním z možných vylepšeních je dvouúrovňový algoritmus poposaný v [9]. V první fázi tohoto algoritmu se nejprve pro každého kandidáta vygeneruje n_0 realizací. Na jejich základě se vyberou ti, 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 upravený 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.

3.5 Iterativní dynamické programování

Iterativní dynamické programování [8] je jední z přístupů k nalezení optimální strategie, která minimalizuje očekávanou ztrátu (2.13). 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.

3.5.1 Diskretizace prostoru

Při hledání optimální strategie $\mu_t(H_t)$ bychom pro přesné vyčíslení očekávané ztráty (3.6) 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 (H_t) a počítat $\mu_t(H_t)$ jen v bodech diskretizace a jinde se uchýlit k interpolaci (popřípadě extrapolaci).

Jakým způsobem efektivně diskretizovat prostor nezávislých proměnných pro aproximativní výpočet očekávané ztráty (3.6) 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 $(H)_{0:N}$. 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 [2].

3.6 SIDP

Metoda stochastického iterativního dynamického programování (SIDP) [11] spočívá v současném použití metody Monte Carlo k získání aproximace pro očekávanou ztrátu a iterativního dynamického programování k nalezení optimální strategie. Pro účely této postačuje základní verze metody Monte Carlo a je proto v následující implementaci SIDP použita. Při použití iterativního dynamického programování se uchýlíme k diskretizovat prostoru hyperstavů a budeme používat interpolaci (popřípadě extrapolaci) napočtených hodnot. Poznamenejme, že díky předpokladu gaussovského rozdělení parametru θ_t, diskretizace vzhledem k T_t znamená diskretizaci vzhledem k $(\hat{\theta}_t, P_t)$.

3.6.1 Algoritmus SIDP

V tomto odílu je schématicky 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 $O(n_{pass} n_{iter} N^2 m n n_g^{\dim H_N})$ (časová náročnost metody Monte Carlo je úměrná vzdálenosti od konce horizontu, proto je časová složitost úměrná druhé mocnině N).

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 $\theta_0, \dots, \theta_{N-1}$ 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

V této kapitole je popsán jednoduchý systém zkoumaný v [1]. Na něm jsou porovnány řídící algoritmy uvedené v předešlé kapitole.

4.1 Popis systému

Výstup systému je popsán jako

$$y_{t+1} = y_t + \theta u_t + v_{t+1} \qquad t = 0, \ldots, N-1, \tag{4.1}$$
$$v_{t+1} \sim N(0, \sigma^2), \tag{4.2}$$

kde rozptyl šumu σ je znám.

O neznámém parametru θ máme v čase t informaci v podobě dostatečné statistiky $T_t = (\hat{\theta}, P_t)$, tvořené střední hodnotou a rozptylem. Předpokládáme nekorelovanost θ s šumem, tedy že

$$\mathrm{Cov}(v_{t+1}, \theta) = 0. \tag{4.3}$$

Ztrátovou funkci volíme kvadratickou, tedy

$$g(y_{0:N}, u_{0:N-1}) = \sum_{t=0}^{N-1} y_{t+1}^2. \tag{4.4}$$

Odhadovací procedurou pro parametr θ je Kalmanův filtr. Pro systém (4.1) má tvar

$$K_t = \frac{u_t P_t}{u_t^2 P_t + \sigma^2} \tag{4.5}$$
$$\hat{\theta}_{t+1} = \hat{\theta}_t + K_t(y_{t+1} - u_t \hat{\theta}_t), \tag{4.6}$$
$$P_{t+1} = (1 - K_t u_t) P_t. \tag{4.7}$$

Hyperstav systému H_t tvoří vektor $(y_t, \hat{\theta}_t, P_t)$. Očekávaná ztráta je

$$J_t(H_t) = \min_{u_t \in U_t} \mathop{\mathrm{E}}_{y_{t+1}, v_t} \left\{ y_{t+1}^2 + J_{t+1}(H_{t+1}) | H_t, u_t \right\}, \qquad t = 0, \ldots, N-1. \tag{4.8}$$

Ta po dosazení z (4.1) a částečném provedení střední hodnoty přejde na tvar

$$J_t(y_t, \theta_t) = \min_{u_t \in U_t} \left\{ (y_t + \hat{\theta}_t u_t)^2 + u_t^2 P_t + \sigma^2 + \mathop{\mathrm{E}}_{y_{t+1}, v_t} (J_{t+1}(y_{t+1}, \theta_{t+1})) | y_t, \theta_t, u_t \right\}. \tag{4.9}$$

4.2 Specifika jednotlivých přístupů

V tomto oddílu jsou popsány některé aspekty algoritmů, které budeme srovnávat, při aplikaci na systém (4.1).

4.2.1 Certainty equivalent control

Očekávaná ztráta (3.5) prejde v

$$J_t(H_t) = \min_{u_t \in U_t} \left\{ \hat{y}_t^2 + J_{t+1}(\hat{H}_{t+1}) | I_t, \theta_t, u_t \right\}. \tag{4.10}$$

Střední hodnota výstupu je

$$\hat{y}_{t+1} = y_t + \hat{\theta}_t u_t \tag{4.11}$$

a rozhodnutí bude tedy

$$\mu_t(y_t, \hat{\theta}_t) = -\frac{y_t}{\hat{\theta}_t}. \tag{4.12}$$

4.2.2 Metoda separace

V první fázi metody separace položíme řídící zásah

$$u_0 = \sqrt{C - \frac{1}{P_0}}. \tag{4.13}$$

Tím se dle (4.5) sníží rozptyl P_0 neznámého parametru θ na $\frac{1}{C}$. Konstanta C by měla být volena dostatečně malá, aby odhad $\hat{\theta}$ pro druhou fázi řízení byl dostatečně blízko skutečné hodnotě parametru θ. Při srovnání jednotlivých algoritmů pokládáme $C = 100$.

4.2.3 SIDP

Dle (4.9) je optimální u_t závislé na $(y_t, \hat{\theta}_t, P_t)$. Při simulaci máme tedy v každém časovém okamžiku t diskretizovat třídimenzionální prostor nezávisle proměnných.

Dle [1] je však před samotnou simulací vhodné přejít k transformaci prostoru $(y_t, \hat{\theta}_t, P_t, u_t)$ do nových proměnných $(\eta_t, \beta_t, \zeta_t, \nu_t)$ dle

$$\eta_t = \frac{y_t}{\sigma} \tag{4.14}$$

$$\beta_t = \frac{\hat{\theta}_t}{\sqrt{P_t}} \tag{4.15}$$

$$\zeta_t = \frac{1}{\sqrt{P_t}} \tag{4.16}$$

$$\nu_t = \frac{u_t\sqrt{P_t}}{\sigma} \tag{4.17}$$

Současně můžeme neurčitost ve výstupu (4.1) reprezentovat jedinou normalizovanou náhodnou veličinou podle

$$s_t = \frac{y_{t+1} - y_t + \hat{\theta}_t u_t}{\sqrt{u_t^2 P_t + \sigma^2}} \sim N(0,1). \tag{4.18}$$

Rovnice pro výstup (4.1) a následující odhad neznámého parametru (4.5) tak přejde v

$$\eta_{t+1} = \eta_t + \beta_t \nu_t + \sqrt{1+\nu^2} s_t \tag{4.19}$$

$$\beta_{t+1} = \sqrt{1+\nu^2}\beta_t + \nu_t s_t \tag{4.20}$$

Přejdeme-li k vhodně upravené očekávané ztrátě, dostaneme

$$V_t(\eta_t, \beta_t, \zeta_t) = \frac{J_t(y_t, \hat{\theta}_t, P_t)}{\sigma^2} \tag{4.21}$$

$$= \min_{\nu_t} \left\{ (\eta_t + \beta_t \nu_t)^2 + \nu_t^2 + 1 + \mathop{\mathrm{E}}_{y_{t+1}, v_t} (V_{t+1}(\eta_{t+1}, \beta_{t+1}, \zeta)) \right\}. \tag{4.22}$$

Nyní spočteme očekávanou ztrátu pro $N-1$.

$$V_{N-1}(\eta_{N-1}, \beta_{N-1}, \zeta_{N-1}) = \min_{\nu_{N-1}} \left\{ (\eta_{N-1} + \beta_{N-1}\nu_{N-1})^2 + \nu_{N-1}^2 + 1 \right\}. \tag{4.23}$$

Derivací získáme optimální zásah jako

$$\nu_{N-1} = -\frac{\eta_{N-1}\beta_{N-1}}{1+\beta_{N-1}^2} \tag{4.24}$$

a očekávanou ztrátu

$$V_{N-1}(\eta_{N-1}, \beta_{N-1}, \zeta_{N-1}) = \frac{\eta_{N-1}^2 + 1}{\beta_{N-1}^2 + 1} \tag{4.25}$$

Protože optimální zásah ν_{N-1} ani očekávaná ztráta V_{N-1} nezávisí na ζ_{N-1}, díky tvaru V_t nebude rovněž optimální zásah ν_t a očekávaná ztráta V_t záviset na ζ_t. Při diskretizaci tedy stačí uvažovat pouze dvoudimenzionální prostor nezávisle proměnných (η_t, β_t).

4.3 Srovnání jednotlivých přístupů

V této sekci jsou porovnány popsané řídící algoritmy na systému (4.1). POPIS EXPERIMENTU

Závěr

Sem prijde zaver

Seznam použitých zdrojů

[1] K. J. Åström and A. Helmersson. Dual control of an integrator with unknown gain. *Computers & Mathematics with Applications*, 12(6):653–662, 1986.

[2] G. Barequet and S. Har-Peled. Efficiently approximating the minimum-volume bounding box of a point set in three dimensions. *Journal of Algorithms*, 38:91–109, 2001.

[3] R. Bellman. *Dynamic Programming.* Princeton University Press, 1957.

[4] D.P. Bertsekas. *Dynamic Programming and Optimal Control, vol. 1.* Athena Scientific, 1995.

[5] AA Feldbaum. *Optimal control systems.* Academic Press, New York, 1965.

[6] J.M. Hammersley and D.C. Handscomb. *Monte carlo methods.* Taylor & Francis, 1964.

[7] R.E. Kalman. A new approach to linear filtering and prediction problems. *Journal of basic Engineering*, 82(1):35–45, 1960.

[8] R. Luus. *Iterative dynamic programming.* CRC Press, 2000.

[9] B.L. Nelson, J. Swann, D. Goldsman, and W. Song. Simple procedures for selecting the best simulated system when the number of alternatives is large. *Operations Research*, 49(6):950–963, 2001.

[10] V. Peterka. Bayesian system identification. *Automatica*, 17(1):41–53, 1981.

[11] A.M. Thompson and W.R. Cluett. Stochastic iterative dynamic programming: a Monte Carlo approach to dual control. *Automatica*, 41(5):767–778, 2005.