

Simulace portfolia – programátorský manuál

1) Programátorský manuál

1.1 Datový model a průtok dat

- **Aktivum** je slovník se základní strukturou:

```
{
  'nazev': str,
  'cena': float,           # počáteční cena
  'vaha': float,          # cílová váha v portfoliu
  'typ': str,             # např. 'akcie', 'dluhopis', 'zlato',
  'hotovost'
  'korelace': float,      # jen pro korelační model
  'mnozstvi': float,      # doplní se po alokaci kapitálu
  'ceny': list[float]     # doplní se během simulace (časová řada)
}
```

- **Portfolio** je list[aktivum].
- **Historie rebalancování** je list[záznam], kde záznam:

```
{
  'den': int,
  'transakce': [
    {'aktivum': str, 'puvodni': float, 'nove': float, 'rozdil': float,
  'poplatek': float}
  ],
  'poplatky_celkem': float,
  'hodnota_portfolia': float
}
```

1.2 main.py

Účel: Orchestrace běhu – načtení vstupů, simulace, exporty, grafy.

Tok:

1. Načte konfiguraci (nacti_konfiguraci).
2. Pro každý soubor portfolia:
 - nacti_portfolio,
 - vypocitej_zakladni_mnozstvi podle pocatecni_hodnota,
 - případně vygeneruje **sdílené ceny** (generuj_sdilene_ceny) a použije je, pokud se aktiva shodují,
 - spustí simulaci (simuluj_portfolio nebo simuluj_portfolio_sdilene),

- vypíše statistiky (vypis_statistiku),
 - exportuje CSV/TXT a kreslí grafy,
 - akumuluje výsledky pro pozdější porovnání.
3. Vykreslí porovnání všech portfolií (PNG i interaktivní HTML).

Poznámky:

- model, sdílená simulace, rebalanční parametry atd. jdou z konfigurace.csv.
- Při nesouladu aktiv u sdílené simulace se loguje varování a jede se nesdíleně.

1.3 portfolio.py

- nacti_portfolio(soubor='portfolio_input.csv')
 - CSV → list aktiv; kontrola součtu vah (± 0.001); ošetření chyb (soubor nenalezen, chybějící sloupec, špatná čísla).
- vypocitej_zakladni_mnozstvi(portfolio, pocatecni_hodnota=1000)
 - Rozdělí kapitál podle vah, pro poslední aktivum doplní **zbytek** kvůli numerice.
 - Nastaví aktivum['mnozstvi'] u všech aktiv.
- vypocitej_vyvoj_portfolia(portfolio)
 - Sumarizace hodnoty přes ceny a množství (není v main.py nutně využito, protože simulace.py vrací vyvoj již přímo).

1.4 konfigurace.py

- nacti_konfiguraci(soubor='konfigurace.csv')
 - Čte CSV parametr, hodnota; konvertuje na bool/float/str (podle obsahu hodnota).
- PARAMETRY_TYPU_AKTIVA
 - Mapa základních ročních parametrů pro typy aktiv:


```
{
    'akcie': {'ocekavany_vynos': 0.08, 'volatilita': 0.15},
    'dluhopis': {'ocekavany_vynos': 0.03, 'volatilita': 0.06},
    'zlato': {'ocekavany_vynos': 0.035, 'volatilita': 0.08},
    'hotovost': {'ocekavany_vynos': 0.01, 'volatilita': 0.0}
}
```
- preved_na_denni(rocni_vynos, rocni_volatilita)
 - Denní drift: $(1 + r)^{(1/252)} - 1$, denní sigma: $\text{vol} / \sqrt{252}$.

1.5 simulace.py

Modely denních cen:

- simuluj_den_nahodny(aktualni_cena, denni_volatilita=0.02)
 - $\text{delta} \sim U(-\text{vol}, +\text{vol})$; $\text{cena} *= (1 + \text{delta})$.
- simuluj_den_typove(aktivum)

- Parametry z PARAMETRY_TYPU_AKTIVA[aktivum['typ']] → denní mu, sigma.
 - $zmena \sim N(\mu, \sigma)$; $cena *= (1 + zmena)$.
- simuluj_den_korelacni(aktivum, index_zmena, denni_volatilita=0.02)
 - $zmena = \rho * index_zmena + (1 - \rho) * U(-vol, +vol)$.

Generátor sdílených cen:

- generuj_sdilene_ceny(portfolio, pocet_dni, model, denni_volatilita)
 - Vrací dict{nazev_aktiva: [cena_0, ..., cena_T]} pro všechna aktiva.
 - U korelačního modelu generuje i vlastní trajektorii „indexu“.

Simulace portfolia:

- simuluj_portfolio(portfolio, cilove_vahy, pocet_dni, rebalancovaci_perioda, zpusob_rebalancovani, tolerance_vahy, transakcni_poplatek, model, denni_volatilita, sdilena_simulace=False)
 - Pro každý den nasimuluje ceny dle modelu.
 - Podmínky rebalancování: perioda / odchylka / kombinace (viz níže).
 - Vrací (vyvoj, historie).
- simuluj_portfolio_sdilene(portfolio, cilove_vahy, ceny_aktiv, rebalancovaci_perioda, zpusob_rebalancovani, tolerance_vahy, transakcni_poplatek)
 - Totéž, ale bere ceny z ceny_aktiv (již vygenerované sdílené trajektorie).

Rebalanční spouštěč:

- Používá rebalancovani.je_odchylka_prilis_velka – porovnává aktuální váhy (počítané z **posledních cen** v ceny[-1]) s cílovými; pokud je rozdíl > tolerance, vrací True.

1.6 rebalancovani.py

- rebalancuj_portfolio(portfolio, den, cilove_vahy, historie, poplatek_sazba=0.005)
 - Vypočte aktuální hodnotu portfolia v **daný den** z ceny[den].
 - Pro každé aktivum spočte **cílovou částku** = celkova_hodnota * cilova_vaha.
 - **Poplatek** = |rozdil_castky| * poplatek_sazba.
 - **Skutečná částka** pro nové množství:
 - nákup: cilova_castka - poplatek,
 - prodej: cilova_castka + poplatek.
 - Přepočte mnozstvi = skutečna_castka / aktualni_cena.
 - Zapiše transakci do historie (včetně součtu poplatků).
- je_odchylka_prilis_velka(portfolio, cilove_vahy, tolerance=0.05)

- Porovnává váhy z **aktuálních posledních cen** vs. cíle.
- `uloz_rebalancovani_do_txt(historie, prefix="")`
 - Uloží přehled všech rebalancí do `vystupy/<prefix>/statistiky/<prefix>_rebalancovani.txt`.

1.7 soubory.py

- Vytváří adresářovou strukturu `vystupy/<prefix>/{statistiky,grafy}`.
- `uloz_transakce_do_csv(historie, prefix) → .../<prefix>_transakce.csv`
- `uloz_vyvoj_portfolia_do_csv(vyvoj, prefix) → .../<prefix>_vyvoj_portfolia.csv`
- `uloz_ceny_aktiv_do_csv(portfolio, prefix) → .../<prefix>_ceny_aktiv.csv`
- `uloz_statistiky_do_csv(vyvoj, prefix, bezrizikova_sazba=0.01) →`
 - `.../<prefix>_statistiky.csv` a `.../<prefix>_statistiky.txt`

Poznámky k formátu:

- Číselné hodnoty jsou formátovány na rozumný počet desetinných míst.
- Ošetřeny hraniční stavy (málo dat apod.).

1.8 statistiky.py

- `vypis_statistiku(vyvoj, bezrizikova_sazba=0.01)` (pokračování)
 - Počítá: celkové zhodnocení, **CAGR**, denní/roční volatilitu (stdev denních výnosů), **Sharpe ratio** ($\text{CAGR} - r_f$)/ vol_rocní , **max drawdown**, min/max hodnotu.
 - Vstupy:
 - `vyvoj: list[float]` — denní hodnoty portfolia (počáteční hodnota i hodnoty po každém dni).
 - `bezrizikova_sazba: float` — roční bezriziková sazba (default 0.01).
 - Výstup: žádný (tisk do konzole). Funkce nevrací hodnoty; metriky se zapisují do CSV/TXT v `soubory.py`.
 - Výjimky: pokud `len(vyvoj) < 2`, funkce pouze vypíše upozornění a vrátí se.

1.9 grafy.py

Účel: Generování grafů pro vizualizaci výsledků simulace. Grafy ukládají obrázky (PNG) a interaktivní HTML (Plotly).

Hlavní funkce (signatury, vstupy, chování):

- `vykresli_ceny_aktiv(portfolio, nazev_souboru='grafy/vyvoj_ceny_aktiv.png', prefix='')`
 - `portfolio: list[dict]` — každý prvek obsahuje minimálně název a ceny (seznam cen po dnech).

- `nazev_souboru`: relativní cesta v rámci podadresáře `grafy/` (default `grafy/vyvoj_ceny_aktiv.png`).
 - `prefix`: složkový prefix pro výstupy/`<prefix>/...`
 - Chování: vykreslí časové řady cen všech aktiv, uloží PNG.
 - Kontroly: ověří, že portfolio není prázdné a že ceny existují.
- `vykresli_vyvoj_portfolia(hodnoty_portfolia, nazev_souboru='grafy/vyvoj_portfolia.png', prefix='')`
 - `hodnoty_portfolia`: `list[float]` — denní celková hodnota portfolio.
 - Vykreslí řadu a uloží PNG.
 - Kontroly: ověří, že `hodnoty_portfolia` není prázdné.
- `vykresli_vyvoj_vah(portfolio, nazev_souboru='grafy/vyvoj_vah.png', prefix='')`
 - Vstupy jako `vykresli_ceny_aktiv` + předpoklad, že portfolio obsahuje `mnozstvi` resp. `historie_mnozstvi` pokud chcete přesné denní váhy.
 - Doporučení: pokud chcete přesnost mezi rebalančními dny, v simulaci ukládejte snapshoty množství po každém dni (`historie_mnozstvi`) a upravte funkci tak, aby je použila.
- `vykresli_drawdown(vyvoj, nazev_souboru='grafy/drawdown.png', prefix='')`
 - `vyvoj`: `list[float]`.
 - Vykreslí procentuální drawdown v čase (graf v %).
- `vykresli_histogram_vynosu(vyvoj, nazev_souboru='grafy/histogram_vynosu.png', prefix='')`
 - Vypočítá denní výnosy (poměrové změny) a kreslí histogram, uloží PNG.
- `vykresli_heatmapu_korelaci(portfolio, nazev_souboru='grafy/heatmapa_korelaci.png', prefix='')`
 - Vytvoří DataFrame denních výnosů pro každé aktivum (pouze pokud existuje alespoň 2 ceny).
 - Vypočítá korelační matici a vykreslí heatmapu pomocí `seaborn.heatmap`.
- `vykresli_realnou_hodnotu_portfolia(vyvoj, inflacni_sazba=0.02, nazev_souboru='grafy/realna_hodnota.png', prefix='')`
 - Diskontuje nominální hodnoty portfolio o roční inflaci (předpokládá 252 obchodních dní).
 - Vykreslí a uloží PNG.
- `vykresli_rolling_volatilitu(vyvoj, okno=63, nazev_souboru='grafy/rolling_volatilita.png', prefix='')`
 - Používá `pandas`: `pct_change` → `rolling(window=okno).std() * sqrt(252)`.
 - Uloží PNG. Okno 63 odpovídá přibližně tříměsíčnímu oknu (63 obchodních dní).
- `vykresli_vyvoj_vice_portfolii(vysledky, prefix='')`
 - `vysledky`: `dict[str, list[float]]` — klíč je název portfolio, hodnota je časová řada.

- Vykreslí několik časových řad do jednoho PNG (vystupy/porovnani/vyvoj_vice_portfolii.png).
- vykresli_vyvoj_vice_portfolii_interaktivne(vysledky, vystup='vystupy/porovnani/vyvoj_portfolii_interaktivne.html')
 - Používá plotly pro vytvoření interaktivního grafu (možnost zoomu/tooltipy) a uloží HTML.

Závislosti a další poznámky: - Modul používá matplotlib, seaborn, pandas a plotly.

Ujistěte se, že jsou nainstalované. - Všude se vytváří adresář vystupy/<prefix>/... pokud ještě neexistuje. - Barvy a styly grafů jsou jednoduché.