MOZA Projekt
Wzmacniacz Kaskodowy 4
(wariant B)

Jakub Półtorak

13 czerwca 2022

Spis treści

Etap 1 **2**

1 Sformułowanie matematyczne zadania optymalizacji **2**
1.1 Optymalizacja jednokryterialna 2
1.2 Optymalizacja wielokryterialna 3

2 Wyznaczenie przybliżenia początkowego rozwiązania **4**

3 Wyznaczanie parametrów roboczych, gładkość funkcji celu, opis kodu **6**
3.1 Wyznaczanie parametrów roboczych 6
3.2 Gładkość funkcji celu i ograniczeń 7
3.3 Opis kodu 8

4 Propozycja rozwiązania numerycznego **9**
4.1 Algorytm, skalowanie 9

Etap 2 **11**

5 Optymalizacja **11**
5.1 Ocena poprawności założeń 11
5.1.1 Funkcje celu i ograniczeń, skalowanie, dokładność i gładkość 11
5.1.2 Wybrane algorytmy optymalizacji 11
5.1.3 Wykorzystane opcje dla algorytmów 12
5.2 Podsumowanie przebiegu optymalizacji 13
5.2.1 Optymalizacja jednokryterialna 13
5.2.2 Optymalizacja wielokryterialna 15

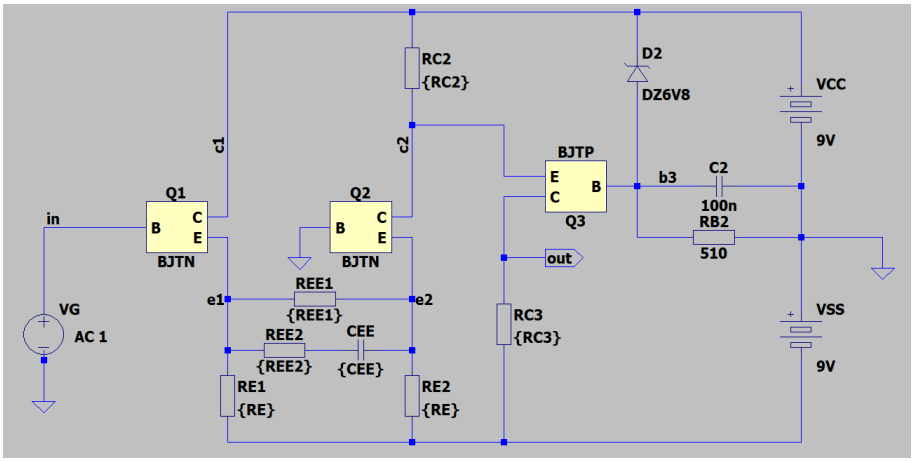
6 Grafiki w wysokiej rozdzielczości. **17**

Etap 1

Opis problemu

Zadanie polega na doborze wartości elementów wzmacniacza tak, aby uzyskać maksymalnie duży iloczyn GBW. Na układ nałożono dodatkowe ograniczenia w postaci minimalnego wzmocnienia dla małych częstotliwości $k_{u0} > 20dB$ ($10\frac{V}{V}$ dla źródła AC o amplitudzie 1 V) oraz minimalnej częstotliwości granicznej $f_g > 200MHz$ (rozumianej jako częstotliwość spadku o 3 dB względem k_{u0}).

Scheamt wzmacniacza przedstawia poniższy rysunek:



Rysunek 1: Schemat optymalizowanego układu.

1 Sformułowanie matematyczne zadania optymalizacji

1.1 Optymalizacja jednokryterialna

Poszukiwane jest minimum funkcji celu:

$$\min_{\mathbf{x}\in\mathbf{R}^+} f(\mathbf{x})$$

p.o.

$$g_i(\mathbf{x}) \leq 0 \quad i = 1..n_g$$

gdzie:

$$f(\mathbf{x}) = -(k_{u0} \cdot f_g)$$

x - wektor zmiennych optymalizowanych:

$$\mathbf{x} = \begin{bmatrix} REE1 & REE2 & RE & RC2 & RC3 & CEE & CG \end{bmatrix},$$

$k_{u0}(\mathbf{x})[dB]$ - wzmocnienie dla małych częstotliwości, rozumiane jako wzmocnienie dla częstotliwości 1 kHz. **Ponieważ w symulatorze LTSPICE do symulacji AC wykrozstywane jest źródło AC o amplitudzie 1V zmieniono zapis ograniczenia (i wszytskich inforamcji dot k_u z jednostki $\frac{V}{V}$ na dB. Wymagane wzmocnienie $10\frac{V}{V} jestrwne 20dB$)**
$f_g(\mathbf{x})[Hz]$ - częstotliwość graniczna, rozumiana jako częstotliwość, dla której wzmocnienie spada o 3 dB względem $k_{u_0}(\mathbf{x})$.

Parametry $k_{u0}(\mathbf{x})$ oraz $f_g(\mathbf{x})$ obliczane są w Matlabie na podstawie surowych danych ($U_{out}^{AC}(x, f)$) zwracanych przez symulator LTSpice.

Dodatkowo, w zadaniu pojawiają się ograniczenia nieliniowe związane z wymaganiami projektowymi:

- $g_1(\mathbf{x}) : -(\frac{k_{u0}(\mathbf{x})}{k_{u_{min}}} - 1) < 0$
  Warunek minimalnego wzmocnienia, $k_{umin} = 20dB$
- $g_2(\mathbf{x}) : -(\frac{f_g(\mathbf{x})}{f_{g_{min}}} - 1) < 0$
  Warunek minimalnej częstotliwości granicznej, $f_{gmin} = 200MHz$. Częstotliwość graniczna f_g obliczana jest jako częstotliwość, dla której wzmocnienie spada o 3 dB względem ku_0.
- $g_3(\mathbf{x}) : b(\mathbf{x}) - b_{max} < 0$
  Ograniczenie podbicia charakterystyki, $b_{max} = 1dB$. Podbicie $b(\mathbf{x})$ rozumiane jest jako różnica między maksymalnym poziomem wzmocnienia a k_{u0} ((czyli $b(\mathbf{x}) = max(k_u(\mathbf{x})) - b_{max}$)).Podbicie jest obliczane w Matlabie.

1.2 Optymalizacja wielokryterialna

Poszukiwane jest minimum funkcji celu:

$$\min_{\mathbf{x} \in \mathbf{R}^+} [f_1(\mathbf{x}), f_2(\mathbf{x})]$$

p.o.

$$g_i(\mathbf{x}) \leq 0 \quad i = 1..n_g$$

gdzie:

$$f_1(\mathbf{x}) = -k_{u0}[dB]$$
$$f_2(\mathbf{x}) = -f_g[Hz]$$

x - wektor zmiennych optymalizowanych:

$$\mathbf{x} = \begin{bmatrix} REE1 & REE2 & RE & RC2 & RC3 & CEE & CG \end{bmatrix},$$

$k_{u0}(\mathbf{x})$ - wzmocnienie dla małych częstotliwości, rozumiane jako wzmocnienie dla częstotliwości 1 kHz.
$f_g(\mathbf{x})$ - częstotliwość graniczna, rozumiana jako częstotliwość, dla której wzmocnienie spada o 3 dB względem $k_{u_0}(\mathbf{x})$.

Parametry $k_{u0}(\mathbf{x})$ oraz $f_g(\mathbf{x})$ obliczane są w Matlabie na podstawie surowych danych ($U_{out}^{AC}(x, f)$) zwracanych przez symulator LTSpice.

Dodatkowo, w zadaniu pojawiają się ograniczenia nieliniowe związane z wymaganiami projektowymi (identyczne jak dla optymalizacji jednokryterialnej):

- $g_1(\mathbf{x}) : -(\frac{k_{u0}(\mathbf{x})}{k_{u_{min}}} - 1) < 0$
  Warunek minimalnego wzmocnienia, $k_{umin} = 20dB$
- $g_2(\mathbf{x}) : -(\frac{f_g(\mathbf{x})}{f_{g_{min}}} - 1) < 0$
  Warunek minimalnej częstotliwości granicznej, $f_{gmin} = 200MHz$. Częstotliwość graniczna f_g obliczana jest jako częstotliwość, dla której wzmocnienie spada o 3 dB względem ku_0.
- $g_3(\mathbf{x}) : b(\mathbf{x}) - b_{max} < 0$
  Ograniczenie podbicia charakterystyki, $b_{max} = 1dB$. Podbicie $b(\mathbf{x})$ rozumiane jest jako różnica między maksymalnym poziomem wzmocnienia a k_{u0} ((czyli $b(\mathbf{x}) = max(k_u(\mathbf{x})) - b_{max})$).Podbicie jest obliczane w Matlabie.

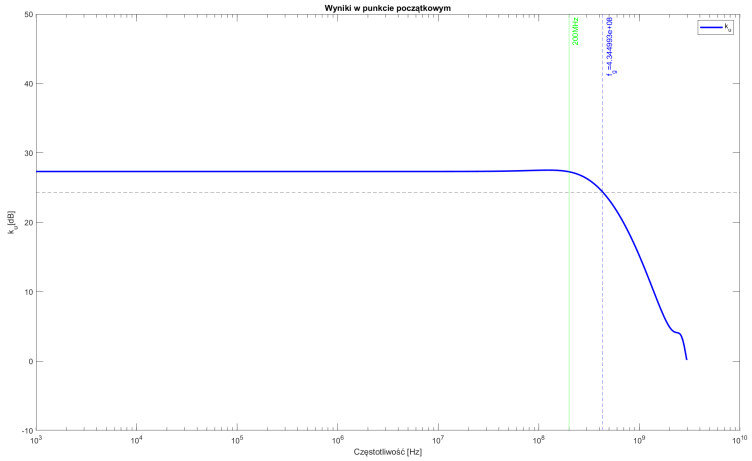
2 Wyznaczenie przybliżenia początkowego rozwiązania

Zgodnie z poleceniem zmodyfikowano domyślne wartości elementów tak, aby uzyskać rozwiązanie spełniające warunek minimalnej częstotliwości granicznej i wzmocnienia. Ostatecznie, po wybraniu wartości, wektor $\mathbf{x}$ wygląda następująco:

$$\mathbf{x} = \begin{bmatrix} 5\Omega & 15\Omega & 320\Omega & 220\Omega & 200\Omega & 45p & 50p \end{bmatrix},$$

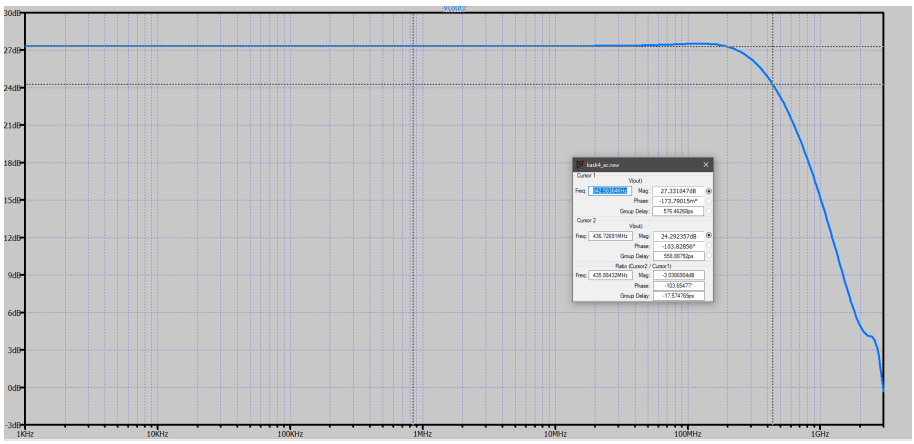
Wyniki w punkcie początkowym można zobaczyć na poniższym wykresie:

Jak widać spełnione są warunki postawione w zadaniu (minimalna wartość wzmocnienia to 20 dB, przy źródle AC mającym 1 V amplitudy) oraz wzmacniacz pracuje prawidłowo (symulacja czasowa wykonana w LTSpice potwierdziła prawidłową pracę układu).



Rysunek 2: Charakterystyka układu w punkcie startowym.

Aby potwierdzić, że Matlab i Spice zwracają te same wyniki przeprowadzono symulację w LTSpice:



Rysunek 3: Charakterystyka układu w punkcie startowym.

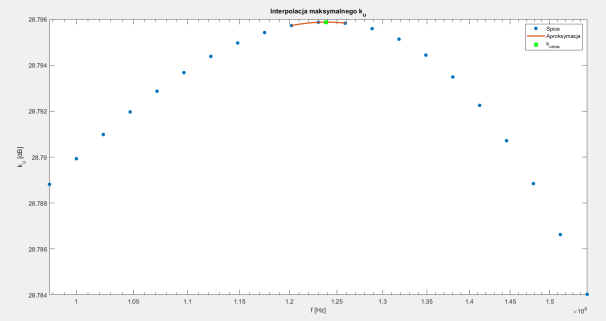
3 Wyznaczanie parametrów roboczych, gładkość funkcji celu, opis kodu

3.1 Wyznaczanie parametrów roboczych

W zadaniu badane są trzy parametry: częstotliwość graniczna, wzmocnienie oraz podbicie charakterystyki.

Podbicie b

Podbicie rozumiane jest jako różnica między wzmocnieniem k_{u0} a maksymalnym wzmocnieniem jakie osiąga charakterystyka ($b(\mathbf{x}) = max(k_u(\mathbf{x})) - b_{max}$). Maksimum charakterystyki jest wyznaczane przez interpolację (wielomian drugiego stopnia). Pozwala to dokładniej ustalić maksymalne wzmocnienie i wygładza funkcję ograniczeń. Efekty interpolacji widać na poniższym wykresie:



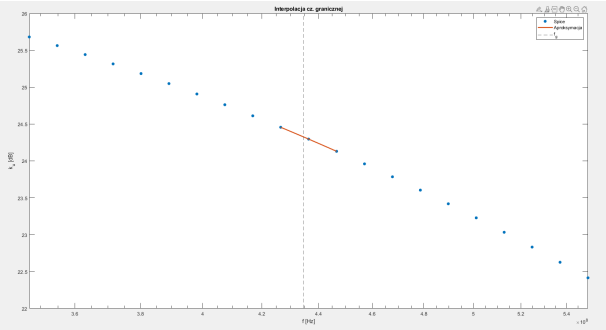
Rysunek 4: Interpolacja maksimum charakterystyki.

Wzmocnienie małoczęstotliwościowe k_{u0}

Wzmocnienie k_u rozumiane jest jako wartość wzmocnienia pozyskana z danych $U_{out}^{AC}(x, f)$ dla częstotliwości 1 KHz.

Częstotliwość graniczna f_g

Częstotliwość graniczna wyznaczana jest jako częstotliwość, dla której wzmocnienie względem k_{u_0} spada o 3 dB. Ponieważ LTSpice zwraca wyniki w postaci punktów, uznano, że wymagana jest interpolacja częstotliwości granicznej. Interpolacja pozwoliła zminimalizować "skoki" w funkcji celu. Do interpolacji wykorzystano wielomian drugiego stopnia. Wynik interpolacji przedstawia poniższy wykres:



Rysunek 5: Interpolacja częstotliwości granicznej.

3.2 Gładkość funkcji celu i ograniczeń

Przyjęto, że funkcja celu, w obrębie odpowiednich wartości elementów układu, jest ciągła. Tak długo, jak w układzie zmieniane są wartości pojemności i rezystancji i nie powodują nieprawidłowej pracy układu zawsze możliwe będzie otrzymanie charakterystyki, która, nawet jeśli nie spełnia wymagań projektowych, daje "sensowne" wartości parametrów roboczych (bez dużych skoków np. wzrost wzmocnienia do 10000 dB).

Funkcje ograniczeń i celu mogą być w niektórych przypadkach niegładkie. Może dojść do takiej sytuacji, gdy aktualny zestaw wartości elementów spowoduje powstanie charakterystyki, dla której niektóre z parametrów roboczych nie są możliwe do obliczenia lub nie mają fizycznego sensu (np. płaska charakterystyka na poziomie 0 dB spowoduje, że nie istnieje częstotliwość graniczna rozumiana wg. zasad zdefiniowanych na początku raportu, funkcja obliczająca zwraca wtedy wartość NaN), lub gdy któreś z ograniczeń zablokuje dany punkt (np. w obliczaniu podbicia wykorzystano operator max). Punkty przejścia między dobrą a złą charakterystyką są właściwie punktami, gdzie funkcja nie jest ciągła (a w niektórych przypadkach także niegładka). Solver fmincon nie spełnia

ograniczeń dla każdej iteracji, przez co teoretycznie może zdarzyć się np. płaska charakterystyka, dla której niektóre parametry nie mają sensu (np. brak spadku częstotliwości o 3 dB), jednak wg. dokumentacji, fmincon jest w stanie wrócić z takich punktów do poprawnej pracy. Przeprowadzone próby wykazały jednak, że problem ten zachodzi bardzo rzadko i ma niewielki wpływ na optymalizację.

W przypadku optymalizacji wielokryterialnej zdecydowano się wykorzystać solver paretosearch (wykorzystujący wewnętrznie algorytm patternsearch), niegładkość funkcji celu i ograniczeń nie ma tutaj więc dużego znaczenia (metoda bezgradientowa).

3.3 Opis kodu

Dołączony katalog z kodem podzielony został na odpowiednie katalogi dla wyników (results, gdzie zapisany jest workspace z Matlaba i plots, wykresy) i plików dla symulacji(spice). W katalogu głównym znajdują się skrypty i m-funkcje. W celu weryfikacji poprawności działania kodu można uruchomić skrypt starting_point.m, który przedstawia wyniki w pkt. początkowym.

Aby uruchomić optymalizację należy uruchomić skrypt main.m. Po optymalizacji wyniki zostaną zapisane do odpowiednich folderów.

Aby nie czekać aż optymalizator zakończy pracę (ok. 5 minut) można wczytać gotowe wyniki za pomocą komendy load('results/latest.mat').

Opis plików

- main.m - główny skrypt realizujący zadanie optymalizacji.
- display_results.m - Skrypt wyświetlający wyniki optymalizacji. Uruchamiany automatycznie po main.m
- starting_point.m - Skrypt obliczający wyniki w pkt. startowym. Do weryfikacji działania funkcji.
- optmization_wrapper.m - funkcja będąca nakładką na optymalizator pozwalającą na współdzielenie między optymalizatorem a funkcjami celu i ograniczeń obliczonych wartości. Funkcja ta wykorzystuje zagnieżdżone funkcje celu i ograniczeń (oraz skalowania). Wszystkie ustawienia (dolne i górne ograniczenia, opcje itp.) znajdują się w tym pliku.
- multiobj_optimization_wrapper.m - Funkcja będąca nakładką na optymalizator wielokryterialny. Podobnie jak w wersji jednokryterialnej steruje całym procesem optymalizacji.
- get_fg.m - Funkcja obliczająca częstotliwość graniczną.
- boost.m - Funkcja obliczająca podbicie charakterystyki.
- extract_results.m - Funkcja odczytująca dane z pliku output_results powstającego przez funckję output_fun.

- modify_params.m - Funkcja modyfikująca parametry w pliku params.inc
- output_fun.m - Funkcja wyjściowa dla optymalizatora.
- run_sim.m - Funkcja uruchamiająca symulator LTSpice.
- get_working_params.m - Funkcja obliczająca podbicie, częstotliwośś graniczną i wzmocnienie.
- LTspice2Matlab.m - Funkcja do odczytu danych z LTSpice.

4 Propozycja rozwiązania numerycznego

4.1 Algorytm, skalowanie

Solver i algorytm optymalizacji jednokryterialnej

Jako solver wykorzystano fmincon z domyślnym algorytmem (Interior Point). Zdecydowano się na wykorzystanie metod gradientowych, ponieważ założono gładkość funkcji dla danych ograniczeń. Po przeprowadzonych próbach optymalizacji uznano, że założenie jest słuszne (nieciągłości jedynie w punktach nie dających wyników o sensie fizycznym, widoczna poprawa parametrów układu, wyraźna zbieżność). Wpływ niegładkości wprowadzanej przez ograniczenie podbicia nie spowodował ani razu przerwania optymalizacji.

Po próbach przeprowadzonych w LTSpice ustalono, że niektóre elementy mają większy wpływ na układ niż inne, jednak, przy różnych kombinacjach wartości, wpływ rożnych elementów jest trudny do przewidzenia. Stwierdzono np. że zmniejszanie wartości REE1 powoduje wzrost wzmocnienia przy spadku pasma, manipulowanie pojemnością CEE wpływa na pasmo, CG na podbicia itp. Ponieważ nie znaleziono jednoznacznej zależności (działanie na kształt algorytmu Gaussa-Seidla) postanowiono nie ograniczać liczby optymalizowanych zmiennych, a jedynie zadbać o odpowiednie ograniczenia kostkowe parametrów.

Solver i algorytm dla optymalizacji wielokryterialnej

W praktyce ciężko określić co jest dobrym rozwiązaniem zadania wielokryterialnego. Przeważnie mamy do czynienia z optymalizowaniem parametrów gdzie poprawa jednego powoduje pogorszenie pozostałych. Aby znaleźć rozwiązanie zadanego problemu zdecydowano się użyć solvera paretosearch (wykorzystującego algorytm patternsearch) do wyznaczenia zbioru Pareto, który powinien dać wybór w kwestii czy bardziej interesuje użytkownika wzmocnienie czy pasmo.

Ponieważ patternsearch jest algorytmem bezgradientowym ewentualny brak gładkości f. celu i ograniczeń nie ma tutaj dużego znaczenia.

Skalowanie

Zarówno wektor wartości elementów jak i funkcja celu zostały przeskalowane.

W przypadku wektora parametrów optymalizowanych zastosowano proste skalowanie do 1 względem punktu startowego $\frac{x}{x_0}$. W badanym przypadku wartości są podobnego rzędu (jednostki zostają dopisane dopiero w funkcji modyfikującej plik z parametrami), jednak dla przejrzystości postanowiono wykonać skalowanie wektora względem wektora startowego.

W przypadku funkcji celu iloczyn wzmocnienia i częstotliwości granicznej sięga rzędu 10^9. Aby usprawnić pracę optymalizatora wyjście z zaimplementowanej funkcji celu jest postaci $-log_{10}(GBW)$.

W przypadku optymalizacji wielokryterialnej częstotliwość graniczna zwracana jest w postaci $-log_{10}(f_g)$.

Wartości wzmocnienia nie przekraczają zakresu od kilku do kilkuset decybeli. Uznano, że nie jest tu konieczne skalowanie.

Etap 2

5 Optymalizacja

5.1 Ocena poprawności założeń

Biorąc pod uwagę wyniki optymalizacji zarówno jedno jak i wielokryterialnej uznano, że założenia z etapu 1 były prawidłowe.

W przypadku optymalizacji jednokryterialnej zauważono poprawę o ok. 68% ($\frac{GBW_{opt}}{GBW_0} = 1.678$) względem punktu startowego. Wszystkie ograniczenia zostały także spełnione, a optymalizator zwraca flagę o możliwości istnienia minimum lokalnego. Omawiane wcześniej założenia dotyczące sytuacji, gdy funkcja celu lub ograniczeń zwróci wartość NaN także okazały się słuszne i optymalizator jest w stanie wrócić do poprawnej pracy z takich punktów.

W przypadku optymalizacji wielokryterialnej wnioski są podobne.

Optymalizator wyznacza granicę Pareto, która układa się w sensowny kształt. Dodatkowo widać, że punkt startowy jest punktem zdominowanym, natomiast te wyznaczone prze algorytm są pareto-optymalne. Punkt wyznaczony jako optymalny w optymalizacji jednokryterialnej także leży na wyznaczonej granicy Pareto.

Mimo, że punktem wyjścia z optymalizatora jest przekroczenie czasu (30 min) uznano, że (badając kształt i wartości na granicy Pareto), że optymalizacja przebiegła poprawnie. Wszystkie ograniczenia także są spełnione.

5.1.1 Funkcje celu i ograniczeń, skalowanie, dokładność i gładkość

W etapie 2 nie wprowadzono zmian dotyczących funkcji celu, ograniczeń, nie zmieniano także skalowania. Założenia dotyczące gładkości funkcji także okazały się słuszne.

Optymalizacja GBW kończy się informacją "Local minimum possible", co oznacza, ze teoretycznie istnieje lepszy punkt niż wynikowy, jednak przyyrsoty dla wartości funkcji celu są mniejsze od ustawionego kroku dla solvera. Teoretycznie można by uruchomić optymalizator z tego punktu jeszcze raz, z mniejszym krokiem, jednak biorąc pod uwagę ograniczenie ze strony wzmocnienia układu (rysunek) uznano, że wyliczony punkt jest wystarczająco bliski optymalnego.

5.1.2 Wybrane algorytmy optymalizacji

Na podstawie wyników oraz przebiegu optymalizacji uznano, że dobór algorytmów był prawidłowy.

W przypadku optymalizacji jednokryterialnej algorytmem był domyślny algorytm gradientowy Interior-Point. Zadanie udało się rozwiązać poprawnie oraz optymalizator zakończył pracę z informacją o możliwości istnienia lokalnego minimum. Jako że powyższy wybór jest wyborem z etapu 1 uznano, że wybór był słuszny i nie zmieniano algorytmu.

W przypadku optymalizacji wielokryterialnej wykorzystano sugerowany w etapie 1 algorytm patternsearch. Biorąc pod uwagę punkty na wyznaczonej granicy Pareto także uznano, ze wybór jest słuszny.

5.1.3 Wykorzystane opcje dla algorytmów

W przypadku solvera dla optymalizacji jednokryterialnej zmieniono następujące opcje:

- "FinDiffRelStep"=1e-2 - zmieniono domyślną długość kroku, gdyż dla niektórych punktów domyślny krok powodował zbyt małe przyrosty funkcji celu.
- "OutputFcn",@output_fun - zaimplementowano własną funkcję wyjściową. Dokładny opis w etapie 1 oraz w kodzie matlaba.

W przypadku optymalizacji wielokryterialnej:

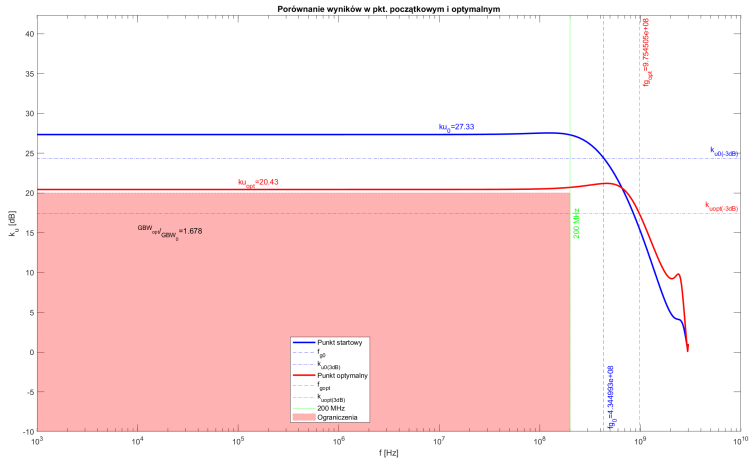
- 'ParetoSetSize' - ustalono maksymalny rozmiar zbioru Pareto. Optymalizator jest w stanie obliczyć dość dużo punktów, jednak zajmuje to bardzo dużo czasu. Uznano, że zbiór o wielkości 50 punktów na pewno pozwoli wywnioskować czy solver działa poprawnie i czy założenia były słuszne. W przypadku jednak, gdyby obliczenie 50 punktów trwało zbyt długo wproadzono także ograniczeń dot. maksymalnego czasu.
- 'InitialPoints' - Solver paretosearch można zainicjować punktami startowymi. W tym przypadku zaporponowano punkt startowy taki sam jak dla optymalizacji jednokryterialnej, gdyż spełniał on wszystkie ograniczenia.
- 'MaxTime' - Ograniczenia dotyczące czasu optymalizacji. Ustawiono na 1800 s, w tym czasie optymalizator wyznaczył ok. 35 punktów na granicy Pareto. Uznano, że zbiór ten dobrze pokazuje działanie optymalizatora oraz daje pogląd co do zachowania układu.

5.2 Podsumowanie przebiegu optymalizacji

5.2.1 Optymalizacja jednokryterialna

Porównanie obiektu przed i po optymalizacji

Wyniki optymalizacji jednokryterialnej zobaczyć można na poniższym wykresie:



Rysunek 6: Porównanie wyników w punkcie startowym i optymalnym.

W trakcie optymalizacji zauważono, że optymalizator zdecydowanie bardziej faworyzuje rozszerzanie pasma zamiast podniesienia wzmocnienia. Prowadzi to do sytuacji, gdy wzmocnienie praktycznie osiąga wartość minimalną, natomiast zyskujemy duże pasmo.

Ograniczenie wzmocnienia jest więc ograniczeniem aktywnym (gdybyśmy pozwolili optymalizatorowi dalej zmniejszać wzmocnienie poszerzałby on dalej pasmo).

Jeżeli użytkownikowi zależy na dużym iloczynie GBW ale dla większego wzmocnienia można albo zwiększyć minimalne wzmocnienie lub nałożyć górne ograniczenie na pasmo (wtedy optymalizator powinien starać się podbijać wzmocnienie). Jednak ponieważ wynik spełnia założenia projektowe postanowiono nie modyfikować procesu optymalizacji.

Warto wspomnieć tutaj także o ograniczeniu dotyczącym podbicia.

W trakcie optymalizacji wielokryterialnej obserwowano przebiegi generowane przez symulator. Zauważono, że wiele przebiegów osiągało wysoką wartość zarówno wzmocnienia jak i pasma jednak podbicie eliminowało takie punkty ze zbioru rozwiązań. Gdyby zwiększyć dopuszczalne podbicie być może uzyskano by lepszy punkt niż optymalny (z optymalizacji wielokryterialnej). To ograniczenie także uznano za aktywne.

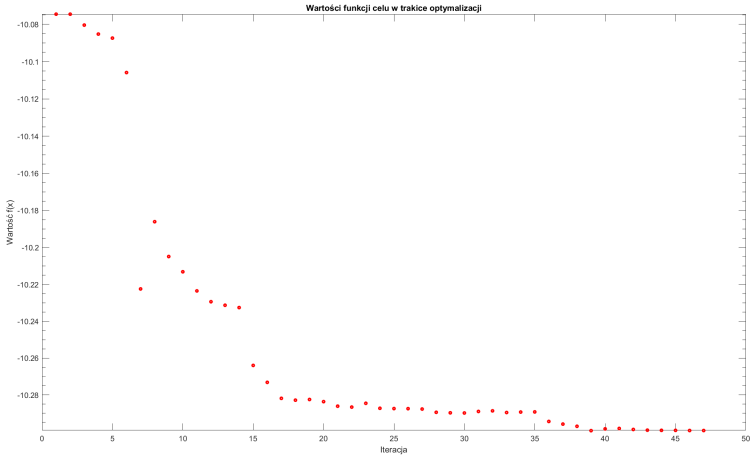
Wartości liczbowe funkcji celu i ograniczeń

Informacje o przebiegu optymalizacji (ze zmiennej "'optim_out"'):

- **Liczba iteracji:** 45
- **Liczba wywołań funkcji:** 478
- **Naruszenie ograniczeń:** 0
- **Wartość funkcji celu w p. optymalnym:** $f(\mathbf{xopt}) = 1.9926 \cdot 10^{10}$
- **Wartość parametrów w p. optymalnym:**
  $x_{opt} = \begin{bmatrix} 9.947\Omega & 1.576\Omega & 536.0\Omega & 254.7\Omega & 157.4\Omega & 19.93p & 125.7p \end{bmatrix}$
- **Powód zakończenia:** "Local minimum possible"

Dodatkowo nałożono ograniczenia kostkowe na optymalizowane wartości. Ograniczenia te zostały dobrane empirycznie po kilku próbach ręcznej optymalizacji układu. Pozwalają one na maksymalnie 10 krotne zmniejszenie zmiennych ($lb = \begin{bmatrix} 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 \end{bmatrix}$), od góry zmienne ogarniczone się weflug wektora: $ub = \begin{bmatrix} 2 & 2 & 5 & 5 & 10 & 10 \end{bmatrix}$ Przebieg wartości funkcji celu zobaczyć

można na poniższej grafice: Funkcja celu zbiega w sposób kwadratowy.



Rysunek 7: Przebieg wartości funkcji celu w trakcie optymalizacji.

- **Czy wskutek użycia optymalizacji zrealizowano wymagania projektowe?**
  Tak, wyniki mieszczą się w zadanych ograniczeniach projektowych.
- **Czy uzyskano widoczną poprawę własności obiektu?**
  Tak, GBW wzrosło o ok 68% względem punkt początkowego. Uznano to za bardzo dobry wynik.

5.2.2 Optymalizacja wielokryterialna

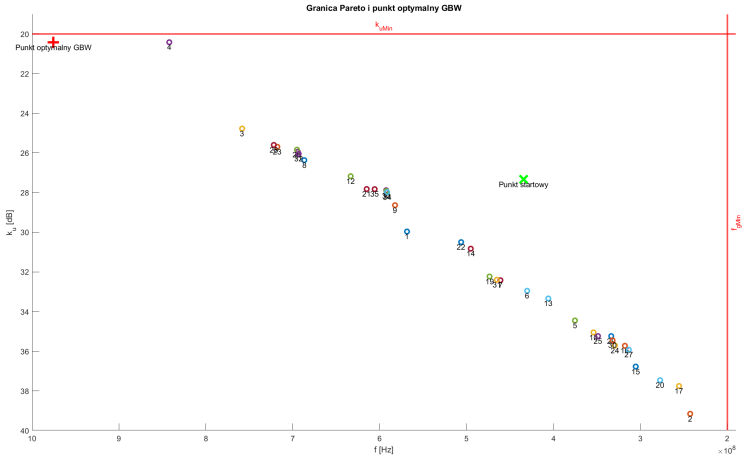
Porównanie obiektu przed i po optymalizacji

Wyniki optymalizacji wielokryterialnej przedstawiono jako zbiór Pareto:

Jak widać każdy z obliczonych punktów jest lepszy zarówno w paśmie jak i wzmocnieniu od punktu startowego. Optymalizator spełnił więc swoje zadanie.

Wartości liczbowe funkcji celu i ograniczeń

- **Liczba iteracji:** 9
- **Liczba wywołań funkcji:** 960
- **Naruszenie ograniczeń:** 0
- **Powód zakończenia:** "Time limit exceeded"

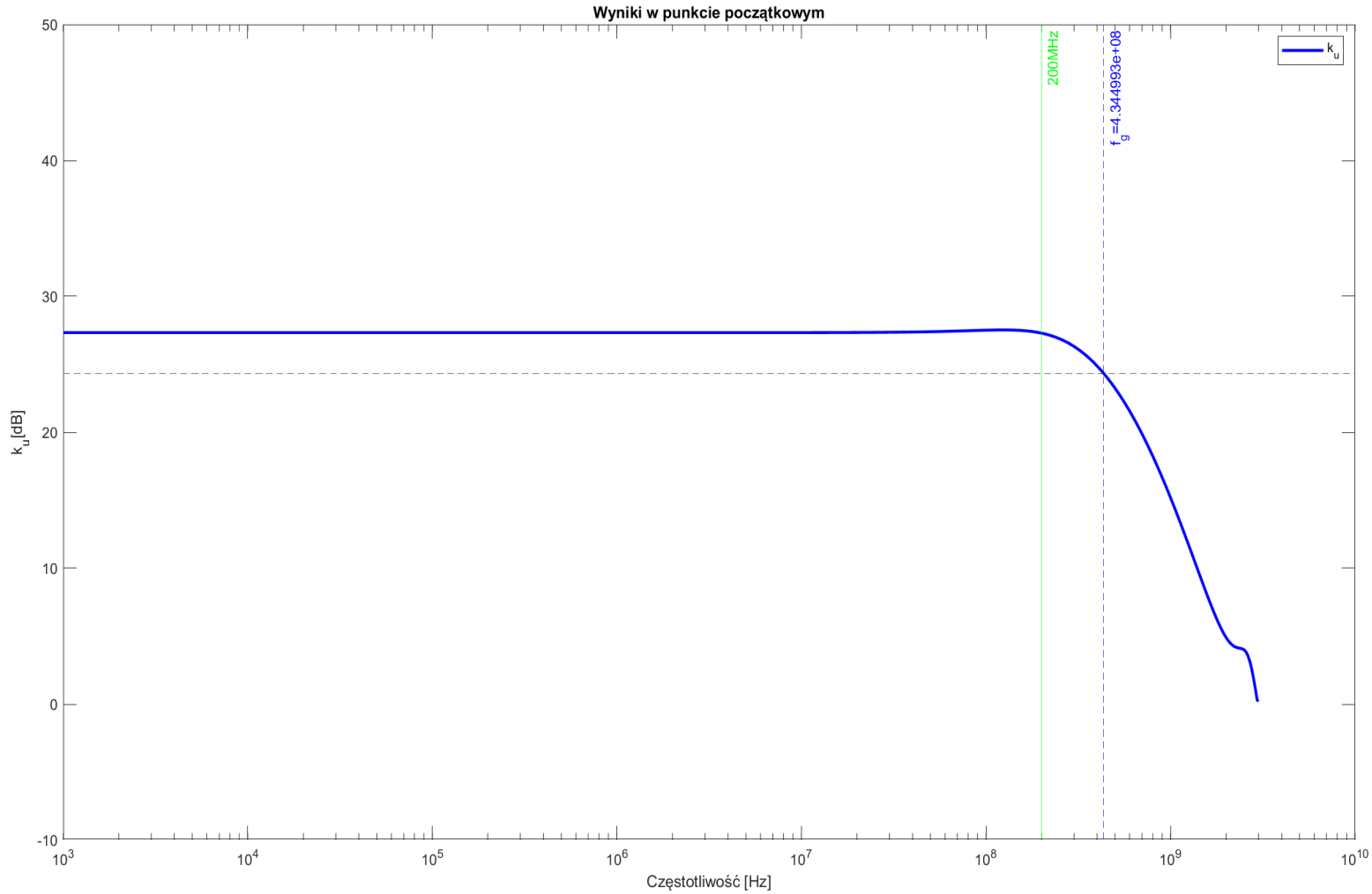


Rysunek 8: Zbiór Pareto dla optymalizacji wielokryterialnej.

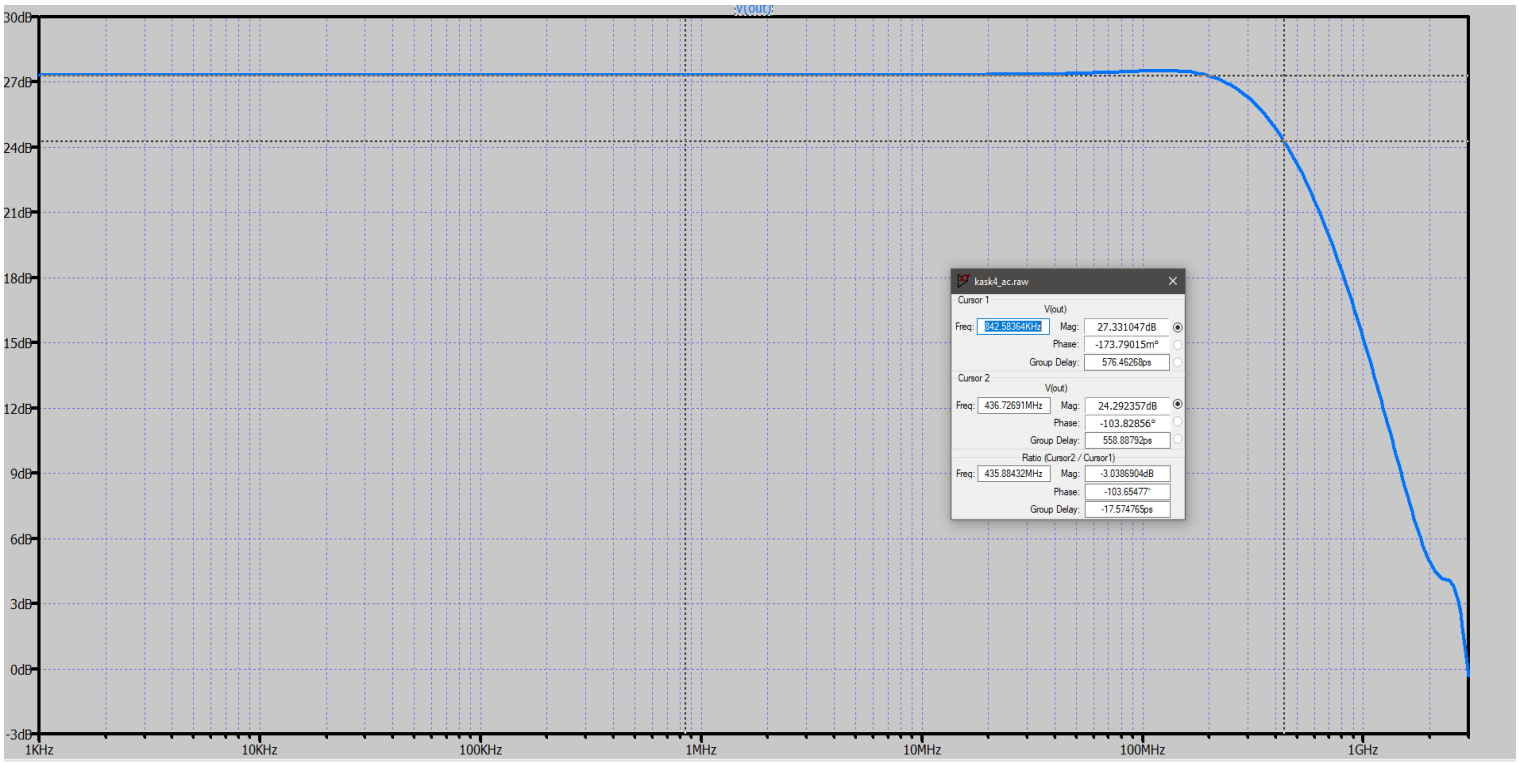
- **Czy wskutek użycia optymalizacji zrealizowano wymagania projektowe?**
  Tak, wszystkie punkty spełniają wymagania.

- **Czy uzyskano widoczną poprawę własności obiektu?**
  W przypadku optymalizacji wielokryterialnej to użytkownik musi ostatecznie wybrać, czy bardziej zależy mu na paśmie czy na wzmocnieniu. Algorytm zwrócił punkty Pareto optymalne, zatem uznano, że optymalizacja powiodła się (każdy punkt lepszy od startowego) i użytkownik może wybrać dowolny punkt z wyznaczonego zbioru Pareto, który zagwarantuje poprawę własności układu.

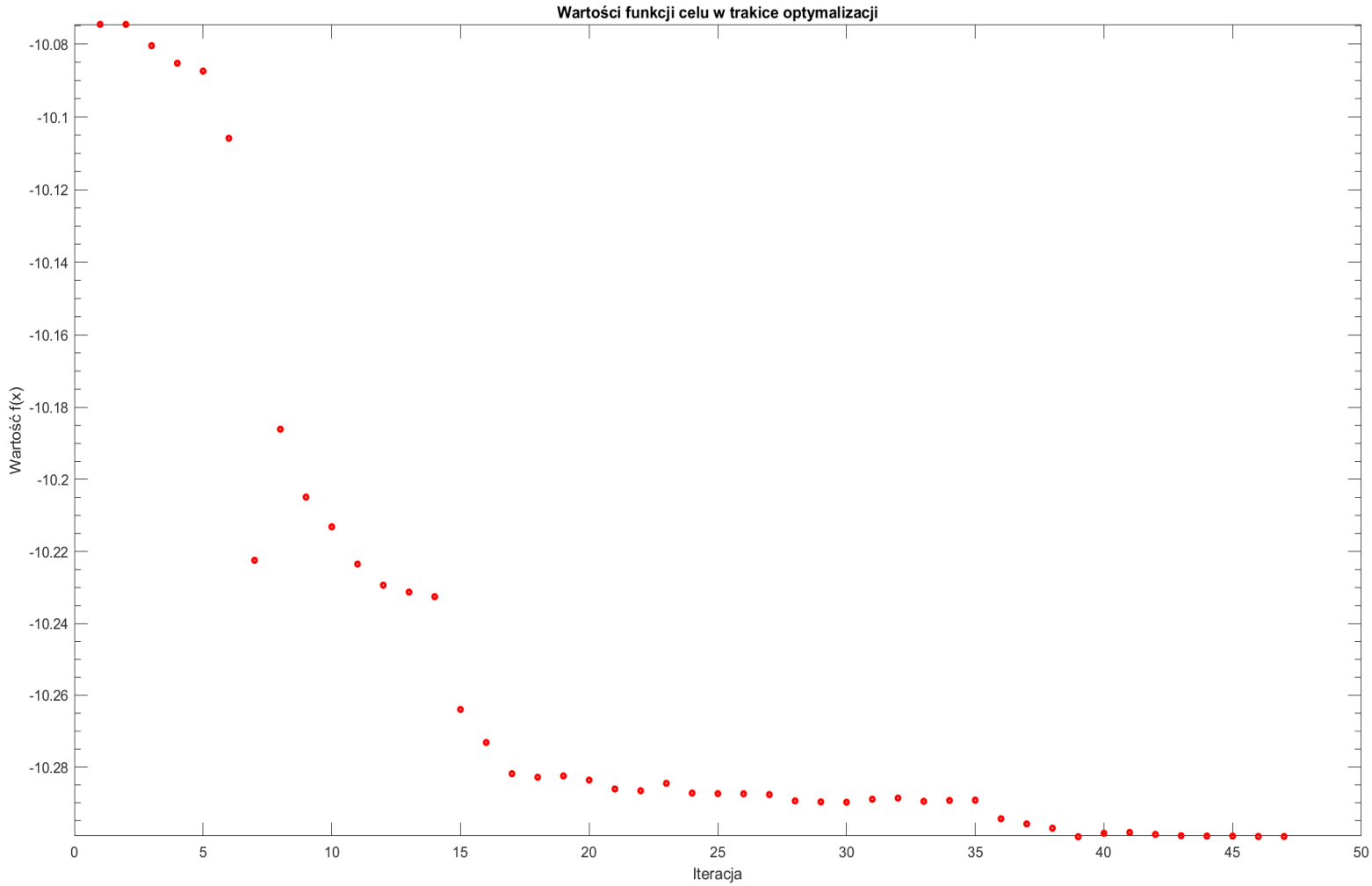
6 Grafiki w wysokiej rozdzielczości.



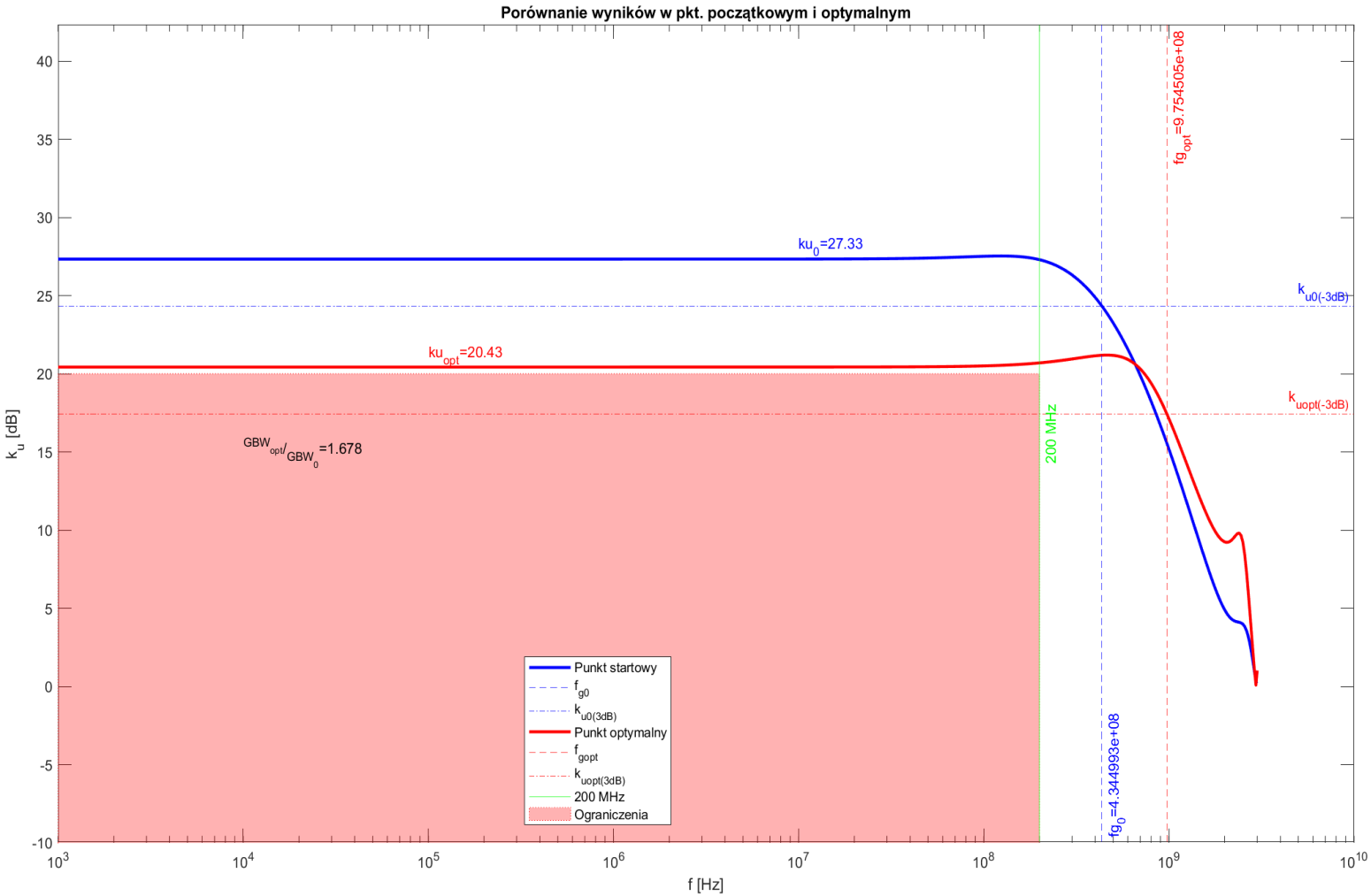
Rysunek 9: Charakterystyka układu w punkcie startowym.



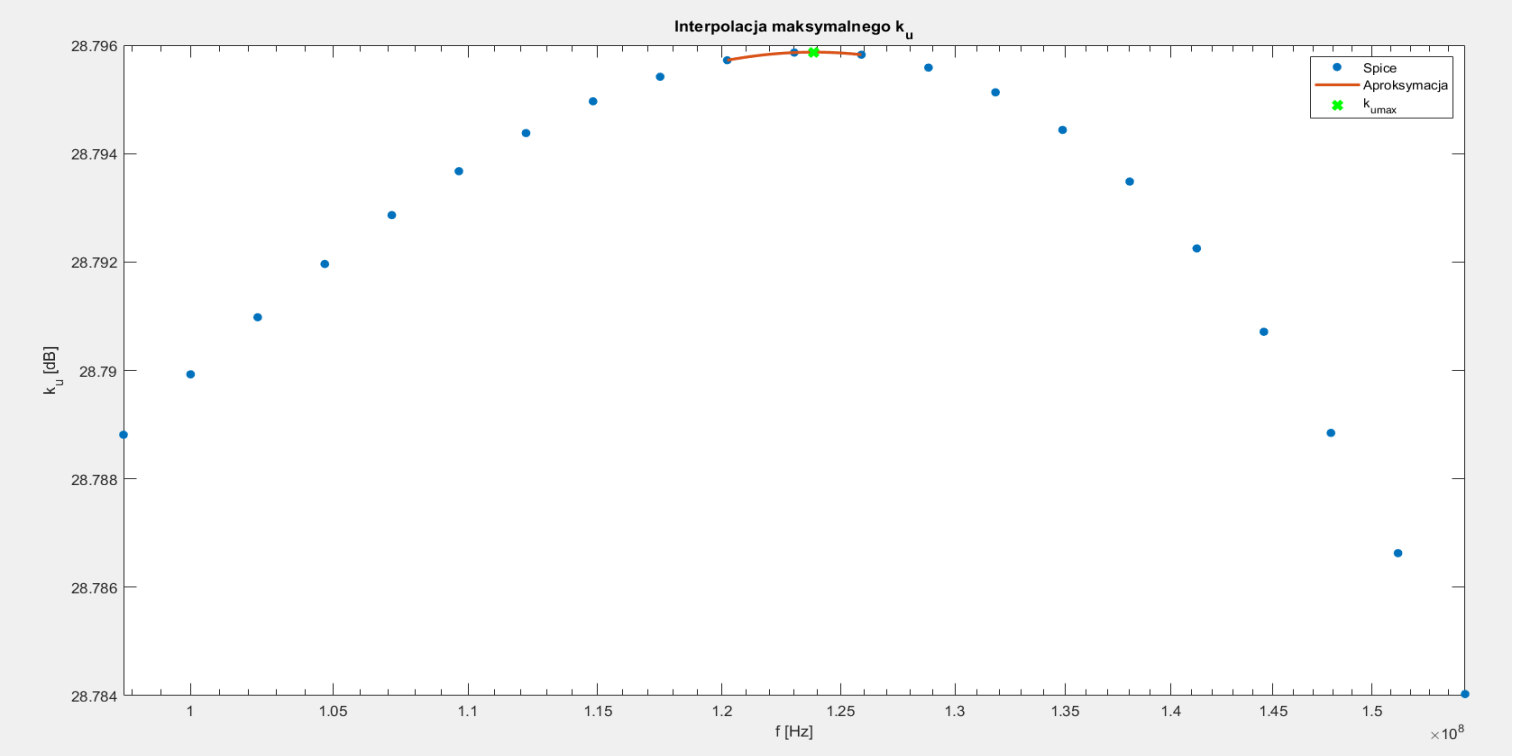
Rysunek 10: Charakterystyka układu w punkcie startowym (LTSpice).



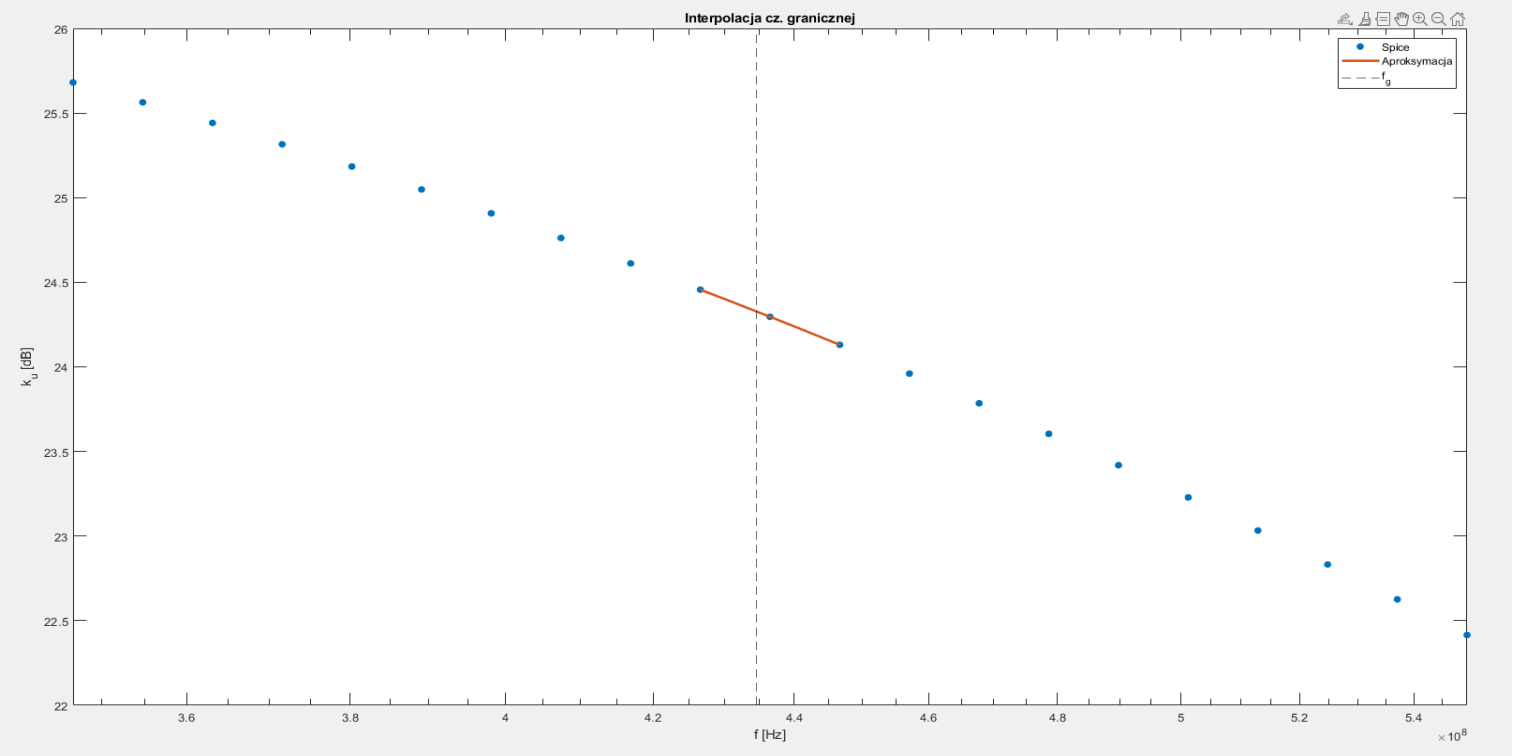
Rysunek 11: Przebieg wartości funkcji celu.



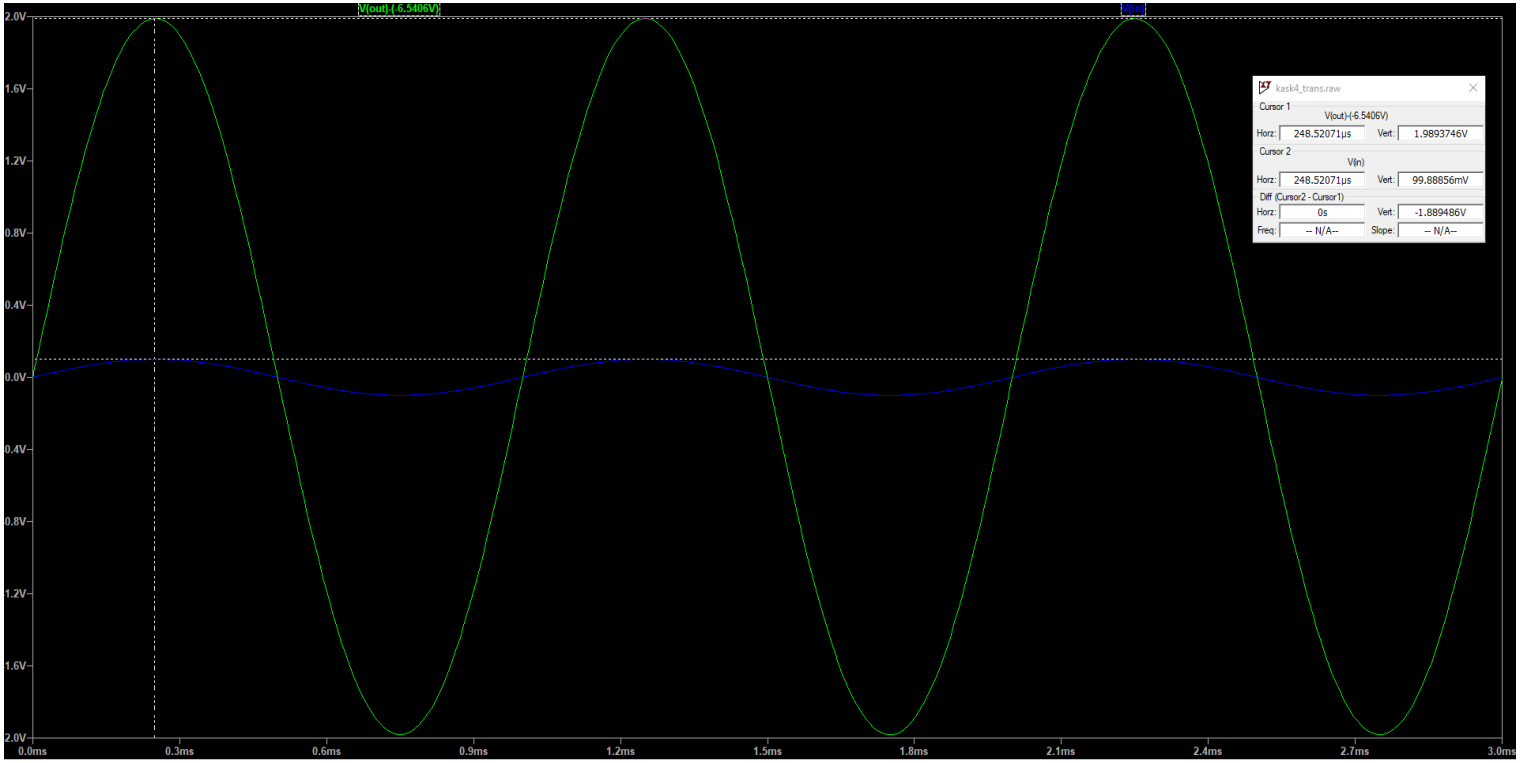
Rysunek 12: Porównanie wyników w punkcie optymalnym i startowym.



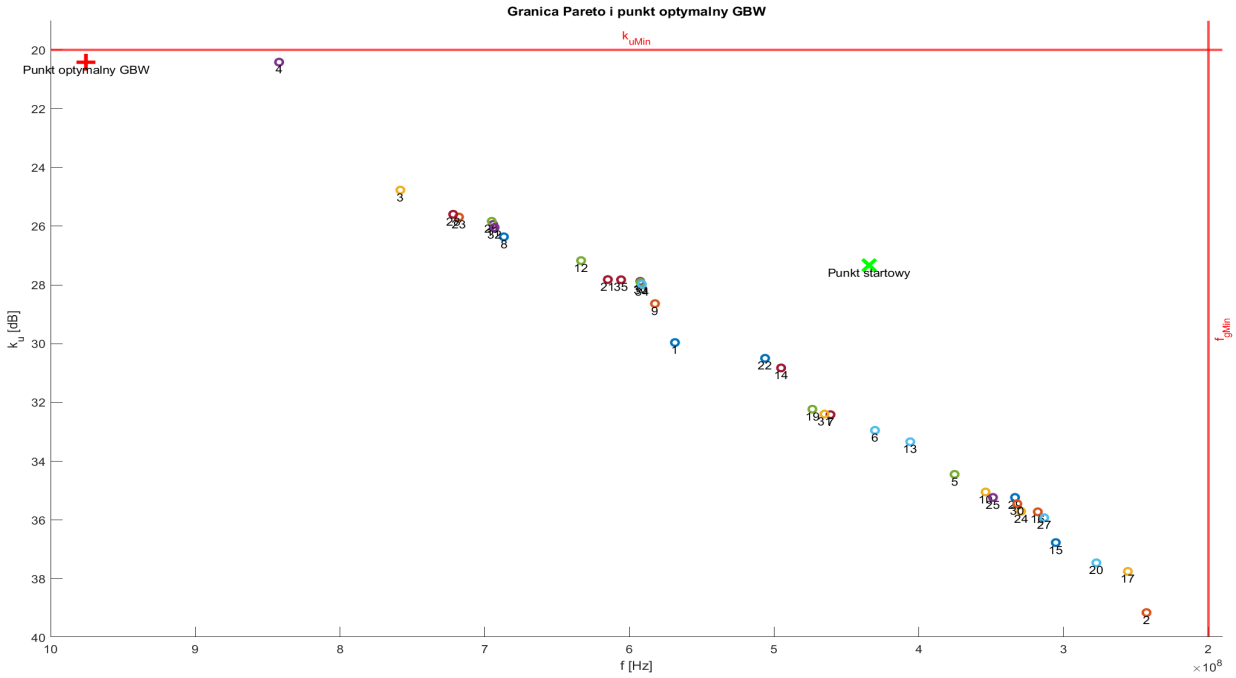
Rysunek 13: Interpolacja maksymalnego wzmocnienia.



Rysunek 14: Interpolacja częstotliwości granicznej.



Rysunek 15: Symulacja czasowa w pkt. optymalnym. Wzmacniacz wzmacnia.



Rysunek 16: Granica Pareto