

Betrachtung eines Bessel-TP --- 6.Mai 2007 Ingenieurbüro Baumann, Dorsten

```
• reset():ta:=time():DIGITS:=32:w:=2*PI*f:
```

die Eingangsdaten

```
• n:=4:fg:=10e3:ue2:=1:
```

die Berechnungen

```
• wg:=2*PI*fg:
```

```
• b:=[fact(2*n-(i-1))/(2^(n-(i-1))*fact(i-1)*fact(n-(i-1))) $  
i=1..n+1]:b0:=b[1]:a0:=b0:
```

```
• delete i:U2U0:=(f)->b0/(a0*(1+1/ue2))*a0/sum(b[i+1]*(I*w/wg)^i,  
i=0..n):
```

```
• U2U0dB:=(f)->20*log(10,abs(U2U0(f))):
```

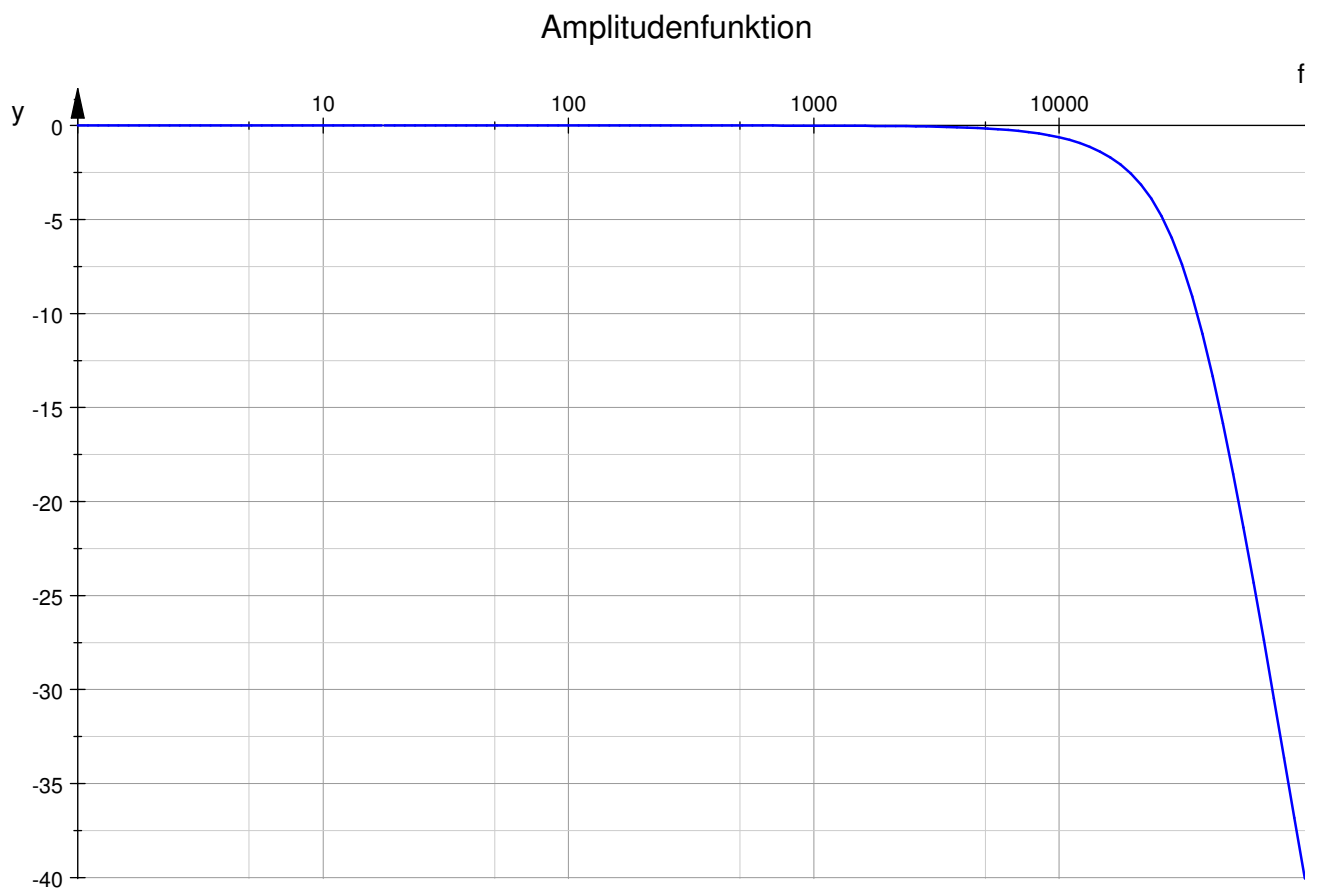
```
• Winkel:=(f)->180/PI*arctan(Im(U2U0(f))/Re(U2U0(f))):
```

```
• Winkel:=(f)->180/PI*arg(U2U0(f)):
```

```
• tg1:=(f)->-diff(Winkel(f),f)/360:
```

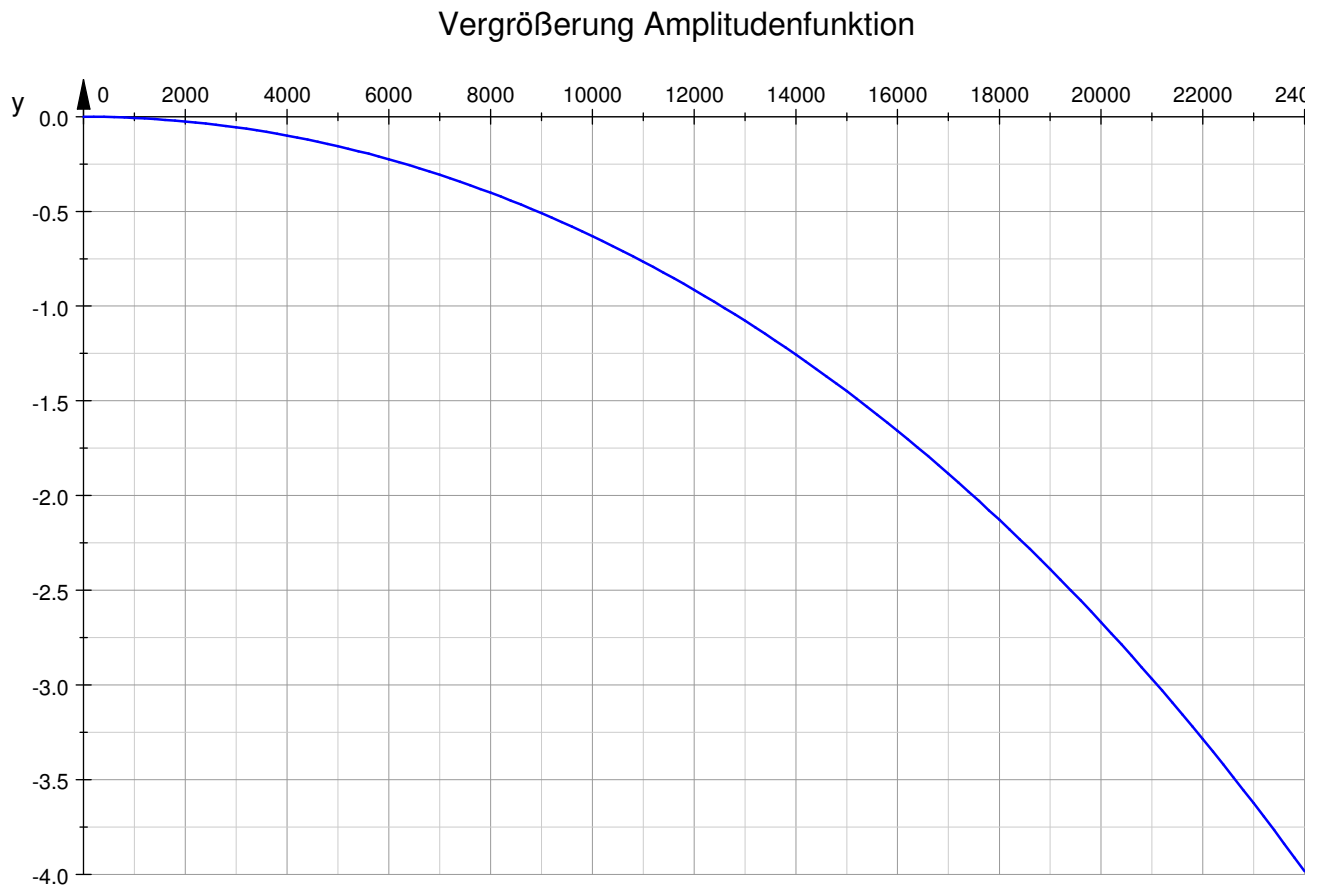
Betrag der Übertragungsfunktion, doppelt logarithmisch

```
• plotfunc2d(U2U0dB(f)+6.02, f=1..10*fg, LegendVisible=FALSE,  
CoordinateType=LogLin,  
GridVisible=TRUE, SubgridVisible=TRUE,  
Height=120*unit::mm, Width=180*unit::mm  
,Header="Amplitudenfunktion"):
```



Ausschnittsvergrößerung aus dem Betrag der Übertragungsfunktion

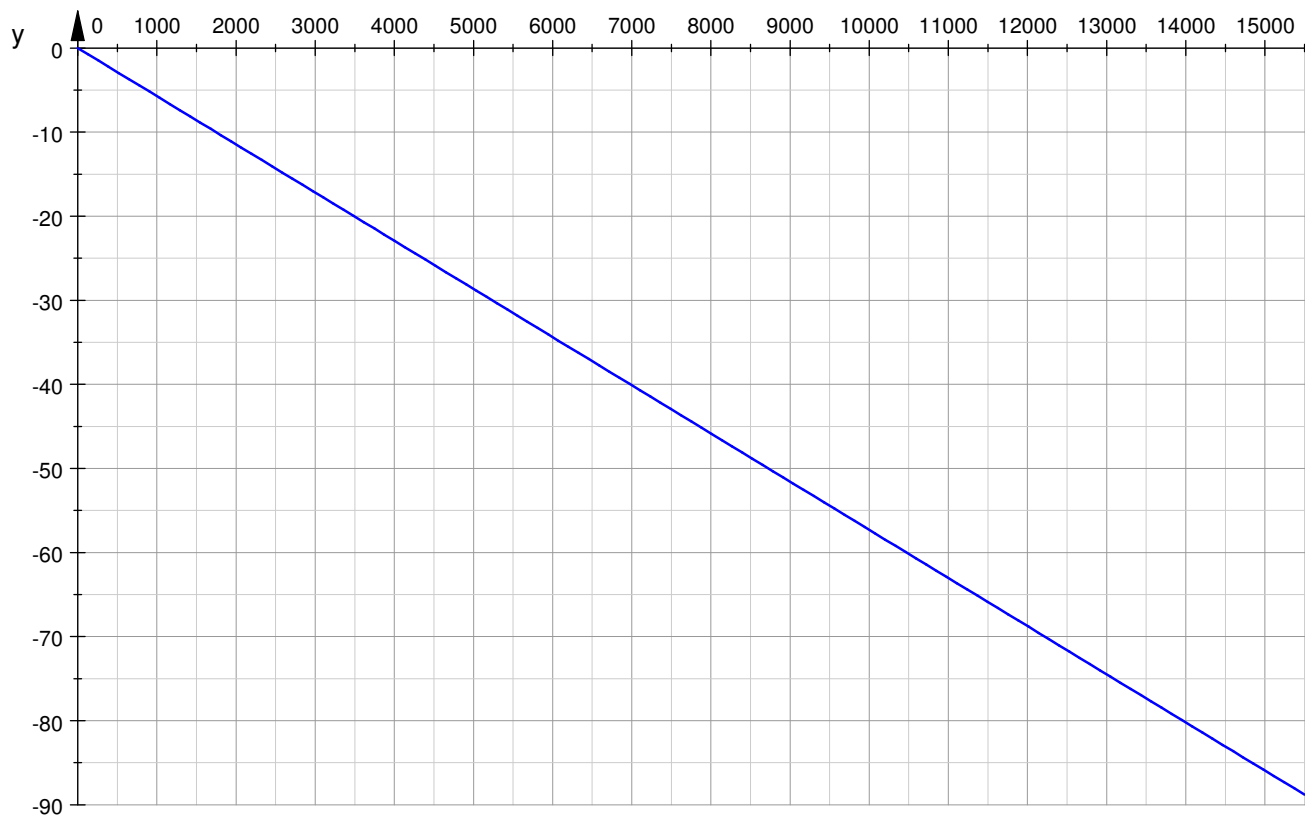
- `plotfunc2d(U2U0dB(f)+6.02, f=0..2.4*fg, LegendVisible=FALSE, GridVisible=TRUE, SubgridVisible=TRUE, Height=120*unit::mm, Width=180*unit::mm, Header="Vergrößerung Amplitudenfunktion"):`



die Phasenverschiebung des Filters

- `plotfunc2d(Winkel(f), f=0..1.55*fg, LegendVisible=FALSE, GridVisible=TRUE, SubgridVisible=TRUE, Height=120*unit::mm, Width=180*unit::mm, Header="Phasenfunktion"):`

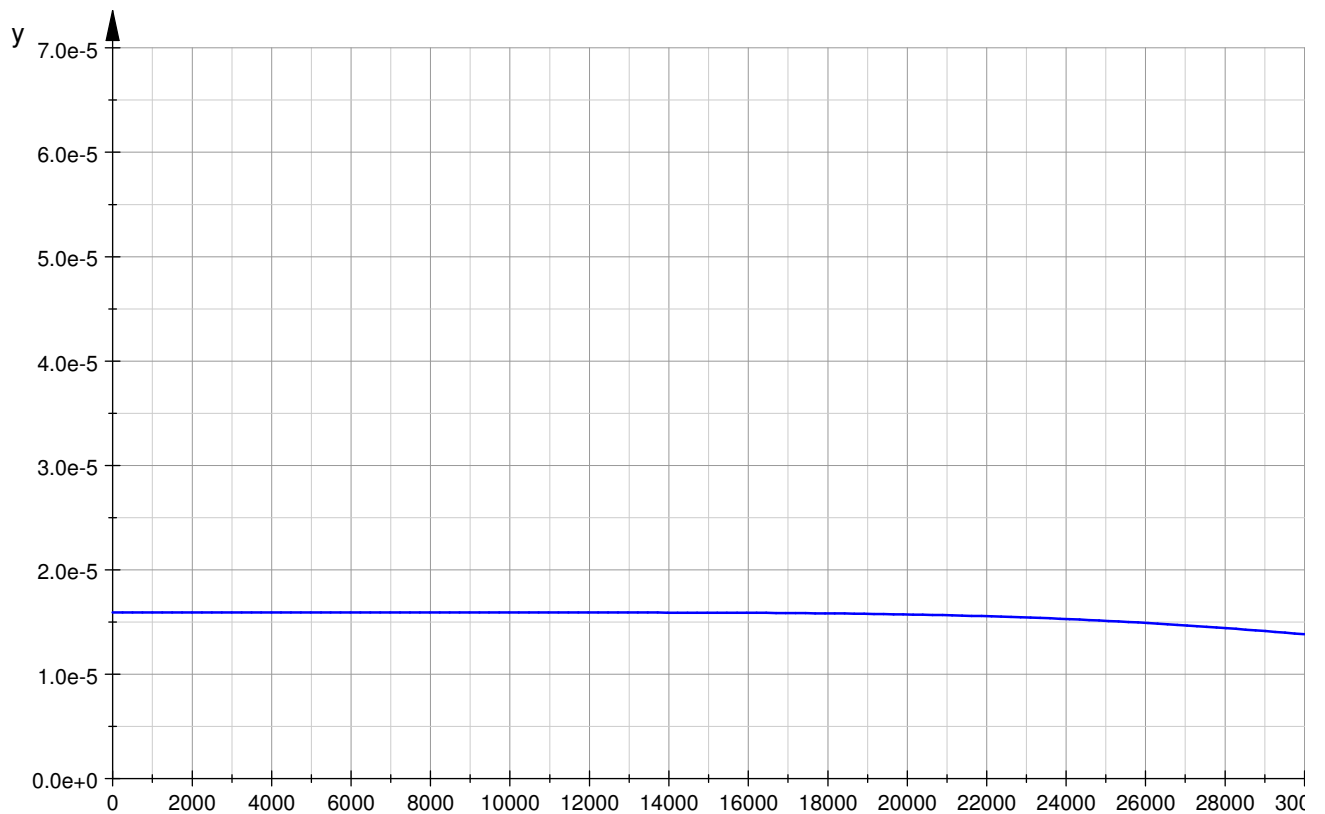
Phasenfunktion



Gruppenlaufzeit aus dem differenzierten Phasenverlauf

- `plotfunc2d(tg1(f), f=0..3*fg, LegendVisible=FALSE, GridVisible=TRUE, SubgridVisible=TRUE, Height=120*unit::mm, Width=180*unit::mm, Header="Gruppenlaufzeit", YRange=0..7e-5):`

Gruppenlaufzeit



das Bessel-Polynom ohne die Grunddämpfung

- `Poly := (F) -> expand (sum (b [i+1] * (F) ^i, i=0..n)) : b;`
`[105, 105, 45, 10, 1]`

- `a0 / Poly (F) ;`

$$\frac{105}{F^4 + 10 \cdot F^3 + 45 \cdot F^2 + 105 \cdot F + 105}$$

die Grunddämpfung durch \ddot{u}^2

- `b0 / (a0 * (1 + 1/ue2)) ;`

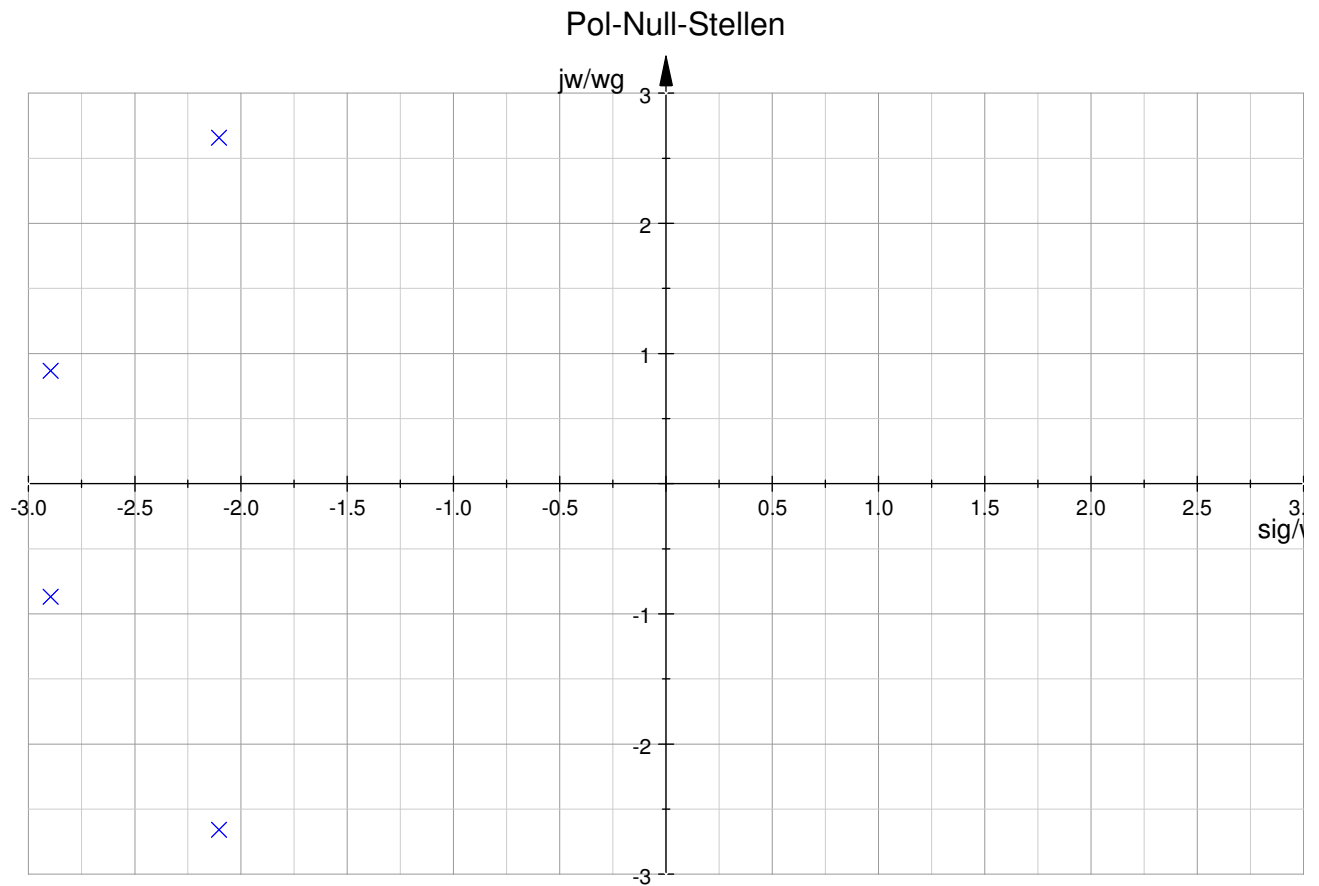
$$\frac{1}{2}$$

- `delete i: Pol := float (solve (Poly (F) = 0, F)) :`
- `delete Liste: for i from 1 to n do`
`PolTab [i] := op (Pol, i) :`
`end_for :`
- `Liste := [[Re (op (op (PolTab, i), 2)), Im (op (op (PolTab, i), 2)), RGB :: Blue] $`
`i=1..n] :`
- `Liste := Liste. [[3, 0, RGB :: White]]. [[0, 3, RGB :: White]]. [[-`

```
3,0,RGB::White]].[[0,-3,RGB::White]]:
```

die Lage der Pol-Nullstellen in der komplexen Ebene

- `plot(plot::PointList2d(Liste, PointStyle=XCrosses, PointSize=2, GridVisible=TRUE, SubgridVisible=TRUE, Scaling=Unconstrained, AxesTitles=["sig/wg", "jw/wg"], Height=120*unit::mm, Width=180*unit::mm, Header="Pol-Null-Stellen")):`



die Pol-Nullstellen

- `PolTab;`

$$\begin{cases} 1 = -2.103789397179627831605606074724 + 2.6574180418567527168583220986318 \cdot i \\ 2 = -2.103789397179627831605606074724 - 2.6574180418567527168583220986318 \cdot i \\ 3 = -2.896210602820372168394393925276 - 0.86723412893450375181897321492012 \cdot i \\ 4 = -2.896210602820372168394393925276 + 0.86723412893450375181897321492012 \cdot i \end{cases}$$

Sprungantwort des Filters mit $u_a(t) = \text{invlaplace}(2/p * T(p))$

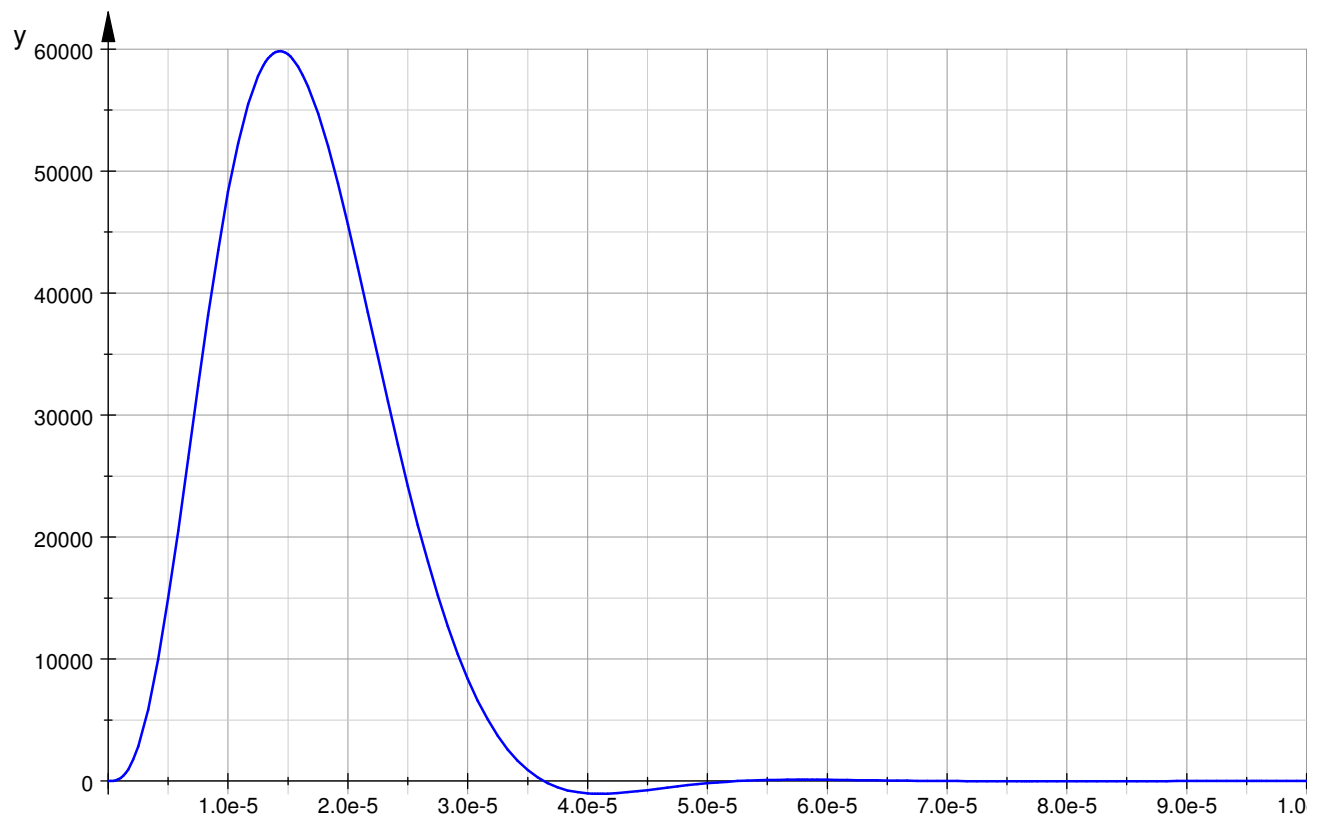
- `prodp:=_mult(p/wg-op(Pol,i) $ i=1..n):`

Impulsantwort des Filters

- `ua:=(t)->Re(transform::invlaplace(b0/(1+1/ue2)*2/prodp,p,t)):`
- `plotfunc2d(ua(t), t=0..1/fg, LegendVisible=FALSE,`

```
CoordinateType=LinLin,
      GridVisible=TRUE, SubgridVisible=TRUE,
      Height=120*unit::mm, Width=180*unit::mm, Header="Impulsantwort"):
```

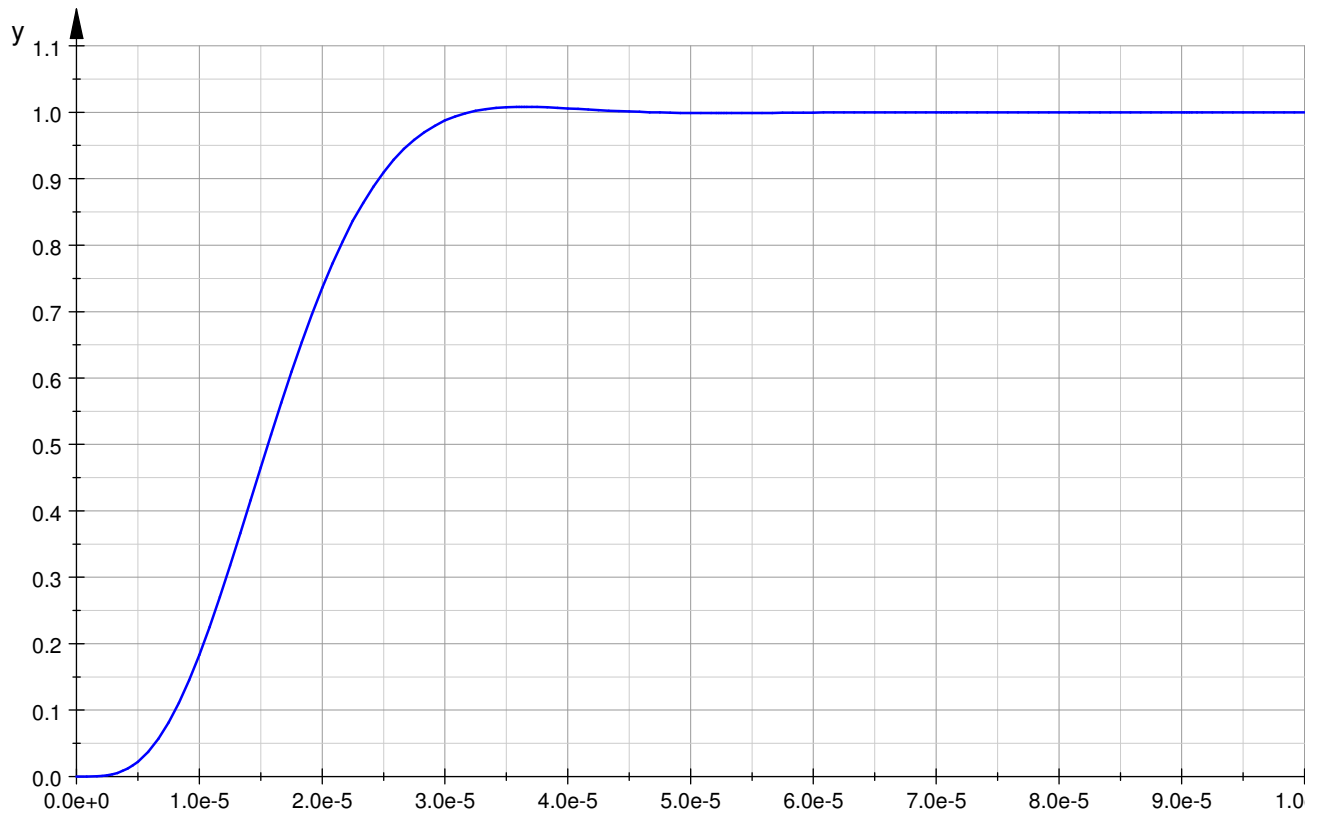
Impulsantwort



Sprungantwort des Filters

- `ua := (t) -> Re(t transform::invlaplace(b0 / (1 + 1/ue2) * 2/p/prodp, p, t)) :`
- `plotfunc2d(ua(t), t=0..1/fg, LegendVisible=FALSE, CoordinateType=LinLin, GridVisible=TRUE, SubgridVisible=TRUE, Height=120*unit::mm, Width=180*unit::mm, Header="Sprungantwort", YRange=0..1.1) :`

Sprungantwort



Suchbereich für Flankendaten

- `anf:=0:ende:=1/2/fg:`

Überschwingen in % bei t in us

- `maximum:=op(numeric::solve(diff(ua(t),t)=0,t=anf..1/2/fg,RestrictedSearch),1):`

- `(ua(maximum)-1)*100;maximum/1e-6;`

0.83541995143491402793358118472718

36.354919486450169423964526030568

t0 für ua(t)=1/2 in us

- `delete t:tx:=op(numeric::solve(ua(t)=1/2,t=anf..maximum,RestrictedSearch),1):tx/1e-6;`

15.580288462336022809616470370504

die Einschwingzeit tau in us und die daraus resultierende Grenzfrequenz in kHz

- `m:=ua'(t):t:=tx:m:=float(m):delete t:yt:=t->1/2-m*(tx-t):`
- `tau:=op(solve(yt(t)=1,t),1)-op(solve(yt(t)=0,t),1):tau/1e-6;float(1/2/PI/tau/1e3);`

16.958092479942069624339684619802

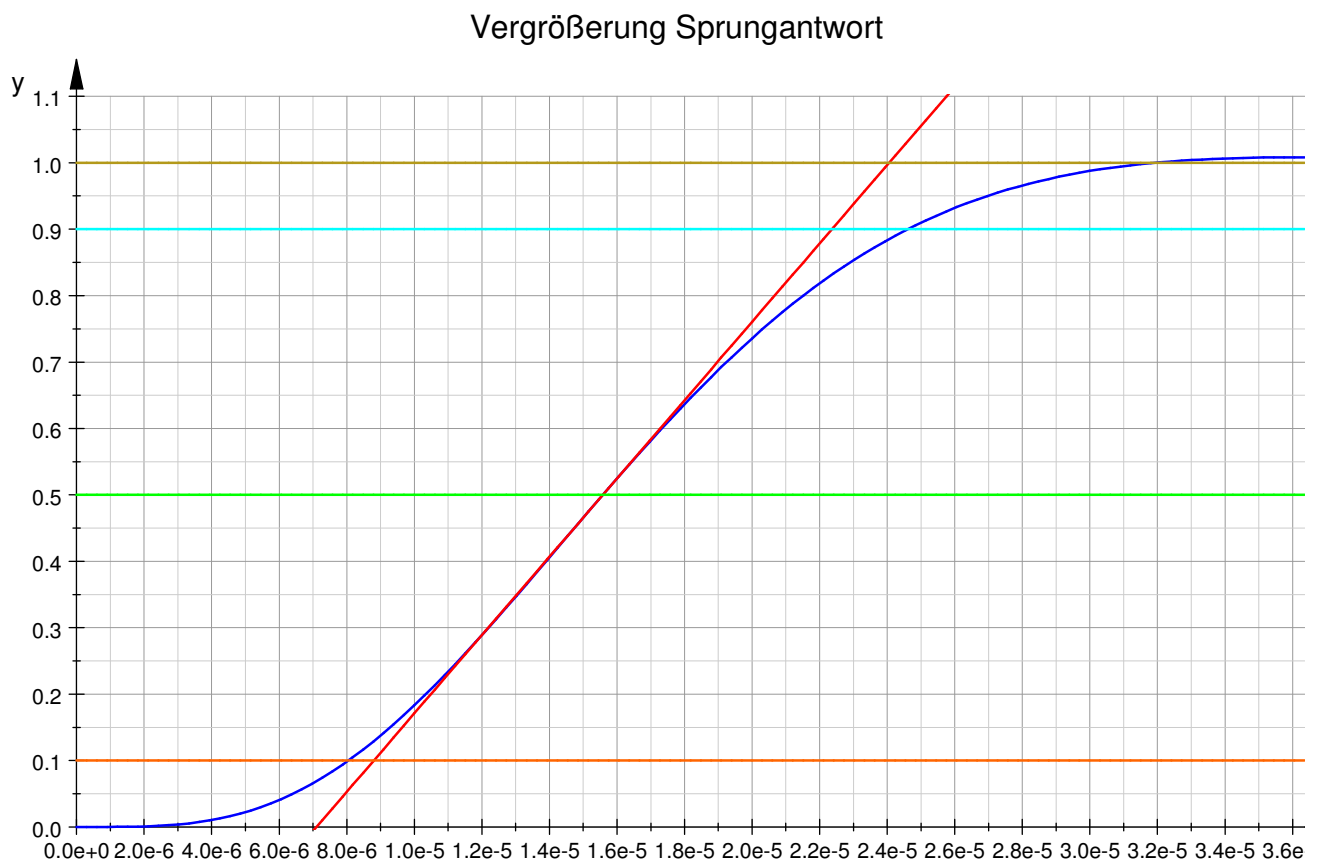
9.385191363954575121543481498692

tr, Rise-Time in us

- `tr:=op(numeric::solve(ua(t)=9/10,t=anf..ende, RestrictedSearch),1)-op(numeric::solve(ua(t)=1/10,t=anf..ende, RestrictedSearch),1):tr/1e-6;`

16.563084815194208343610079938117

- `plotfunc2d(ua(t), yt(t), 1/2, 1, 1/10, 9/10, t=0..maximum, LegendVisible=FALSE, CoordinateType=LinLin, GridVisible=TRUE, SubgridVisible=TRUE, Height=120*unit::mm, Width=180*unit::mm, Header="Vergrößerung Sprungantwort", YRange=0..1.1):`



CPU-Zeit in Sekunden und in Minuten

- `te:=time():float((te-ta)/1000);float((te-ta)/1000/60);`

0.953

