Vergleich verschiedener Ein- und Mehrschrittverfahren zur Lösung von gewöhnlichen Differentialgleichungen

Konvergenzordnungen

- Euler Verfahren Konvergenzordnung 1
- Verb. Euler Verf. Konvergenzordnung 2
- Heun Verf. Konvergenzordnung 2
- X-Schritt AB Verf. Konvergenzordnung X
- Mittelpunkt Verf. Konvergenzordnung 2

```
def normal(p_ziel_func,p_GDL,p_step,p_goal_number,var,overwrite_start=None):
	[Deklaration relevanter Variablen]
	[Erstellen von Verfahrenslisten ]
	[Berechnung von Zielfunktionswerten]
	[Auslesen plot-relevanter Daten (Funktionsname etc.) und erste Definition des Gesamtplots]
	if normale_ver_erlaubt:
		for verfahren in einschritt_verfahren:
			[Verfahrensaufruf, Approximation der Zielfunktion, Erstellen einer Ergebnisliste, Subplot]
		for verfahren in mehrschritt_verfahren:
			[...]
			[zusätzlich: Unterscheidung zwischen verschiedenen Verfahren (Vermeidung überzähliger Initialwerte)]
	if scipy_ver_erlaubt:
		[...]
	[Subplots Fehlerübersicht]
	return x_array, alle_verfahren, pd.DataFrame.from_dict(werte_dict)
[x Werte, Fehler dict, Anzahl Funktionsaufrufe]
```

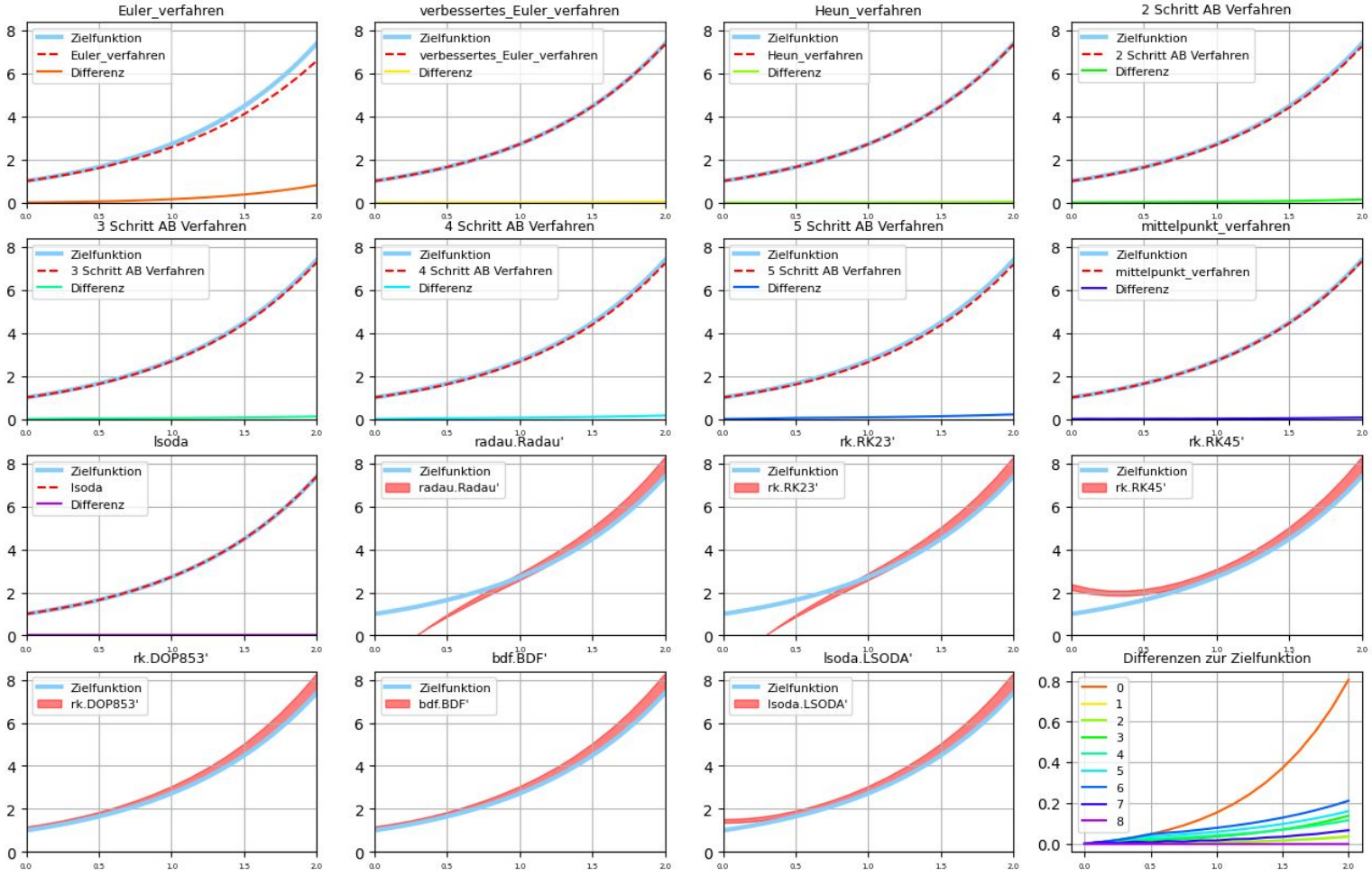
```
for verfahren in einschritt_verfahren:
    verfahren_name = str(verfahren)
    verfahren_name = verfahren_name[10:verfahren_name.find("at 0")-1]
    fig.add_subplot(sqrt_of_l, sqrt_of_l, index)
    esv_res_list,aufrufe = esv.generelle_einschritt_verfahren(start, x_array, iter, step, Gdl, verfahren)
    werte_dict[verfahren_name] = [aufrufe]
    plot_ziel_func(x, ziel_func)
    plt.plot(x_array, esv_res_list, "--", label=verfahren_name, c="r")
    diff_list = []
    for i, value in enumerate(esv_res_list):
        diff_list.append(ziel_func_arr[i] - value)
    line = plt.plot(x_array, diff_list, label="Differenz")
    line[0].set_color(cm(index/3*3/num_diff_colors))
    plot_details(verfahren_name, y_ticks, c_zoom)
    alle_verfahren[verfahren_name] = diff_list
    index += 1
```

```
def generelle_mehrschritt_verfahren(start_value, x_values, number_of_iterations, step, func, verfahren, var, steil_abl):
	anzahl_aufrufe=0
	start_value = [start_value]
*	res_list,anzahl_aufrufe = generate_starting_values(start_value,x_values,var,step,func,anzahl_aufrufe)
*	for i in range(number_of_iterations - (var-1)):
*		parameter_array = []
*		x_array = []
*		for j in range(var):
*			parameter_array.append(res_list[i+(var-j-1)])
*			x_array.append(x_values[i+j])
		res,anzahl_aufrufe = verfahren(step,x_array,parameter_array,func,number_of_iterations,var,anzahl_aufrufe)
		res*= steil_abl
		res_list.append(res)
	return res_list,anzahl_aufrufe
```

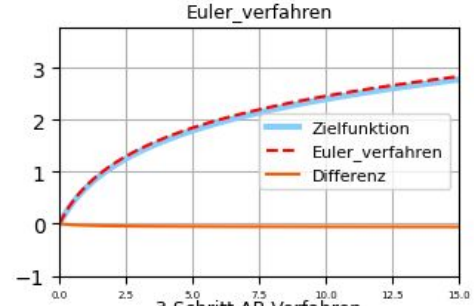
```
def Euler_verfahren(step, x, val, func, anzahl_aufrufe=0):
       return val + step * func(x,val), anzahl_aufrufe + 1


def verbessertes_Euler_verfahren (step, x, val, func, anzahl_aufrufe=0):
       temp_x = x + 1/2 * step
       temp_val = val + 1/2 * step * func(x,val)
       return val + step * func(temp_x,temp_val), anzahl_aufrufe + 2


def Heun_verfahren(step, x, val, func, anzahl_aufrufe=0):
       temp_val = val + step * func(x,val)
       temp_x = x + step
       return val + 1/2 * step * ( func(x,val) + func(temp_x,temp_val) ), anzahl_aufrufe + 3
```

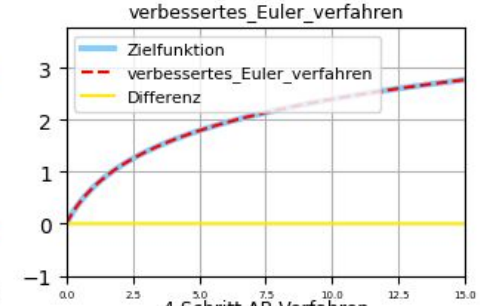
```
def Adams_Bashforth_Verfahren(step, x_values, values, func, number_of_iterations, var, anzahl_aufrufe=0):
	iter = str(var)
	koeffizienten = k_Adams_Bashforth[iter]
	res = 0
	for i in range(len(koeffizienten)-1):
		res += koeffizienten[i+1] * func(x_values[var-i-1], values[i])
		anzahl_aufrufe += 1
	return values[0] + koeffizienten[0] * step * res, anzahl_aufrufe

def mittelpunkt_verfahren(step, x_values, values, func, number_of_iterations, var, anzahl_aufrufe=0):
	return values[1] + 2 * step * func(x_values[0], values[0]), anzahl_aufrufe + 1

def generate_starting_values(curr_list, x_values, iter, step, func, anzahl_aufrufe=0):
	if len(curr_list) == iter:
		return curr_list,anzahl_aufrufe
	result, anzahl_aufrufe = esv.Euler_verfahren(step, x_values[len(curr_list)-1], curr_list[len(curr_list)-1], func, anzahl_aufrufe)
	curr_list.append(result)
	return generate_starting_values(curr_list, x_values, iter, step, func, anzahl_aufrufe)
```

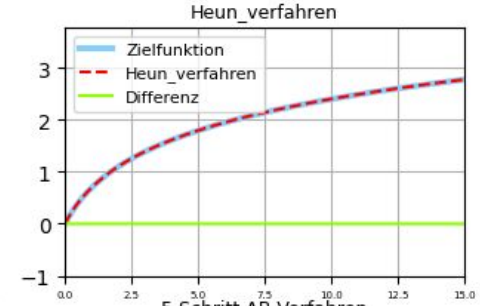
- ```
  def euler_function(x, t=None):
  ```
  - ```
    return np.exp(x)
    ```
- ```
  def euler_f_ableitung(x, val, t=None):
  ```
  - ```
    return val
    ```
- ```
  def n_euler_function(x,t=None):
  ```
  - ```
    return -1 * np.exp(-x)
    ```
- ```
  def n_euler_f_ableitung(x, val, t=None):
  ```
  - ```
    return -val
    ```
- ```
  def log_function(x):
  ```
  - ```
    return np.log(x+1)
    ```
- ```
  def log_ableitung(x,val,t=None):
  ```
  - ```
    return 1/(x+1)
    ```
- ```
  def sin_func(x,val=None):
  ```
  - ```
    return sin(x)
    ```
- ```
  def cos_func(x,val=None,t=None):
  ```
  - ```
    return cos(x)
    ```
- ```
  def tang_func(x,val=None):
  ```
  - ```
    return tan(x)
    ```
- ```
  def tang_func_ableitung(x, val, t=None):
  ```
  - ```
    return 1 + pow(val,2)
    ```
- ```
  def banalt(x,val=None):
  ```
  - ```
    return x**2
    ```
- ```
  def banal_abl(x,val,t=None):
  ```
  - ```
    return 2*x
    ```

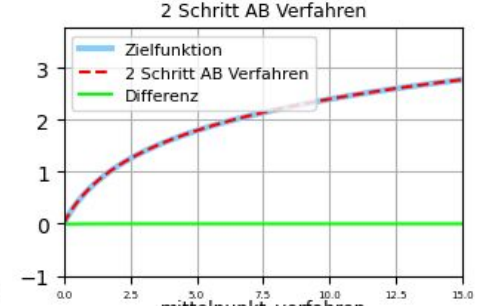

euler_function
Euler_verfahren
Zielfunktion
Euler_verfahren
Differenz
verbessertes_Euler_verfahren
Zielfunktion
verbessertes_Euler_verfahren
Differenz
Heun_verfahren
Zielfunktion
Heun_verfahren
Differenz
2 Schritt AB Verfahren
Zielfunktion
2 Schritt AB Verfahren
Differenz
3 Schritt AB Verfahren
Zielfunktion
3 Schritt AB Verfahren
Differenz
4 Schritt AB Verfahren
Zielfunktion
4 Schritt AB Verfahren
Differenz
5 Schritt AB Verfahren
Zielfunktion
5 Schritt AB Verfahren
Differenz
mittelpunkt_verfahren
Zielfunktion
mittelpunkt_verfahren
Differenz
lsoda
Zielfunktion
lsoda
Differenz
radau.Radau'
Zielfunktion
radau.Radau'
rk.RK23'
Zielfunktion
rk.RK23'
rk.RK45'
Zielfunktion
rk.RK45'
rk.DOP853'
Zielfunktion
rk.DOP853'
bdf.BDF'
Zielfunktion
bdf.BDF'
lsoda.LSODA'
Zielfunktion
lsoda.LSODA'
Differenzen zur Zielfunktion
0
1
2
3
4
5
6
7
8

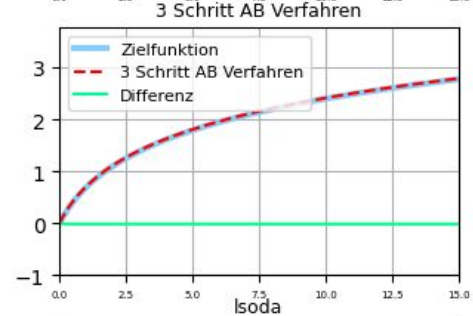
log_function

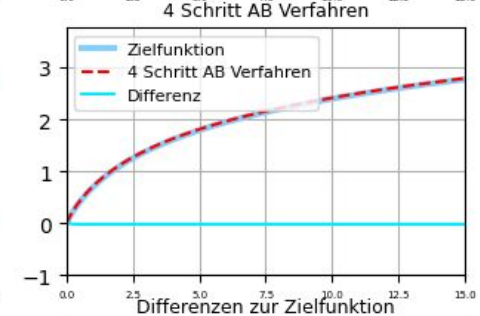


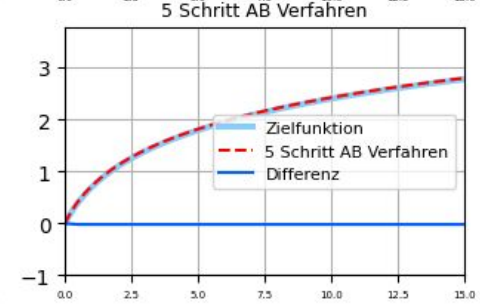


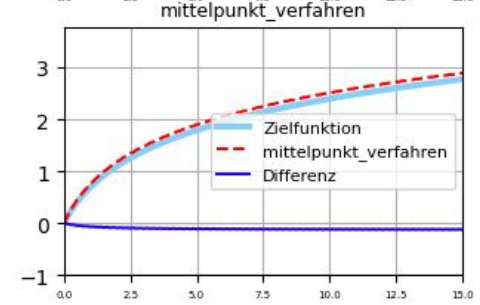


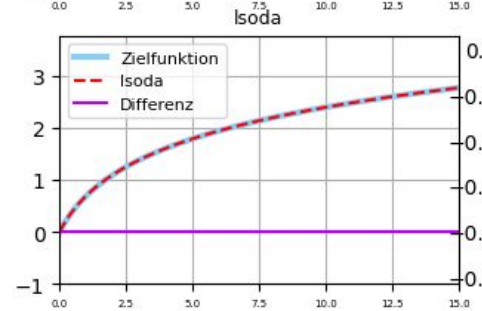


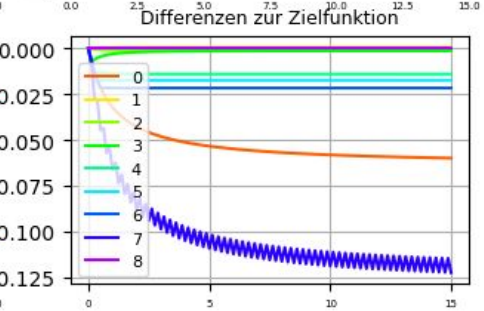


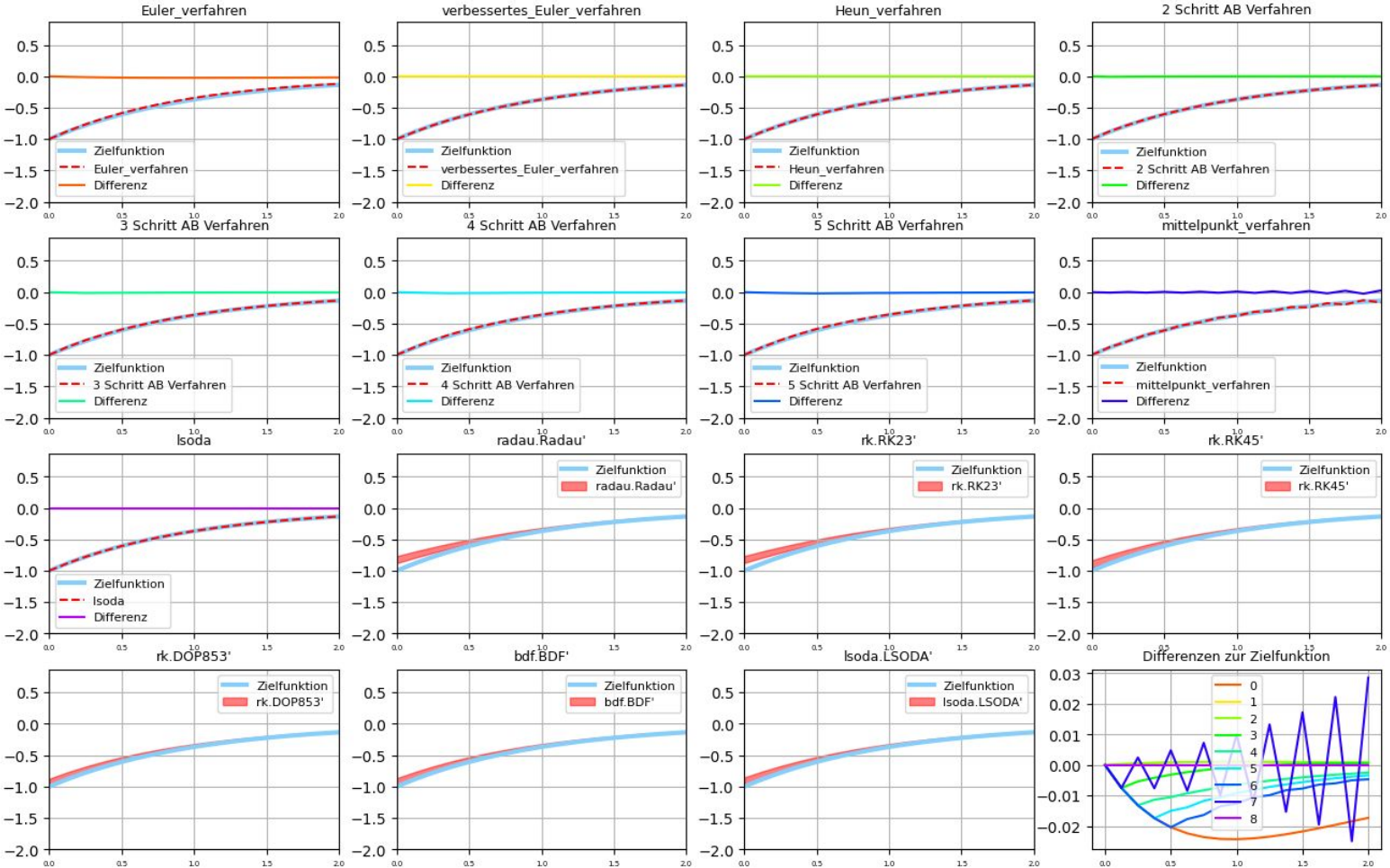




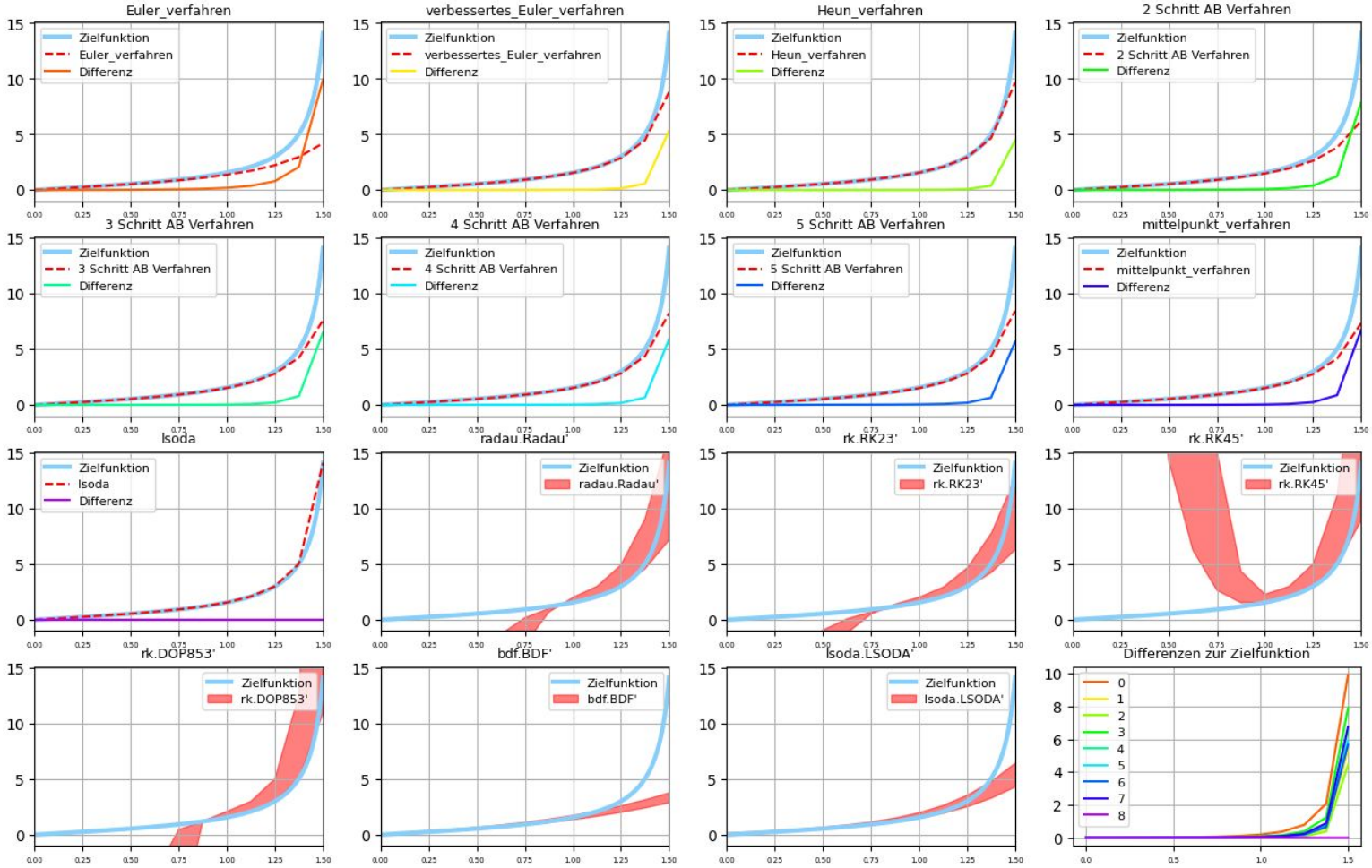




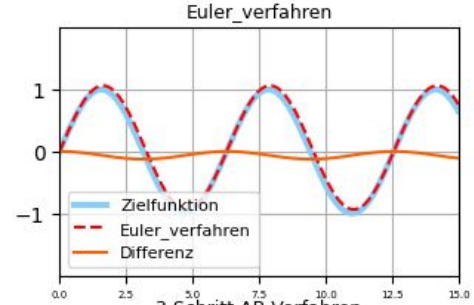


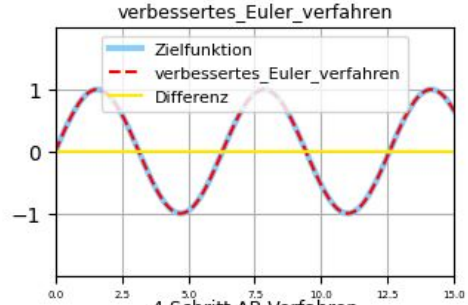

n_euler_function
Euler_verfahren
Zielfunktion
Euler_verfahren
Differenz
verbessertes_Euler_verfahren
Zielfunktion
verbessertes_Euler_verfahren
Differenz
Heun_verfahren
Zielfunktion
Heun_verfahren
Differenz
2 Schritt AB Verfahren
Zielfunktion
2 Schritt AB Verfahren
Differenz
3 Schritt AB Verfahren
Zielfunktion
3 Schritt AB Verfahren
Differenz
4 Schritt AB Verfahren
Zielfunktion
4 Schritt AB Verfahren
Differenz
5 Schritt AB Verfahren
Zielfunktion
5 Schritt AB Verfahren
Differenz
mittelpunkt_verfahren
Zielfunktion
mittelpunkt_verfahren
Differenz
lsoda
Zielfunktion
lsoda
Differenz
radau.Radau'
Zielfunktion
radau.Radau'
rk.RK23'
Zielfunktion
rk.RK23'
rk.RK45'
Zielfunktion
rk.RK45'
rk.DOP853'
Zielfunktion
rk.DOP853'
bdf.BDF'
Zielfunktion
bdf.BDF'
lsoda.LSODA'
Zielfunktion
lsoda.LSODA'
Differenzen zur Zielfunktion
0
1
2
3
4
5
6
7
8

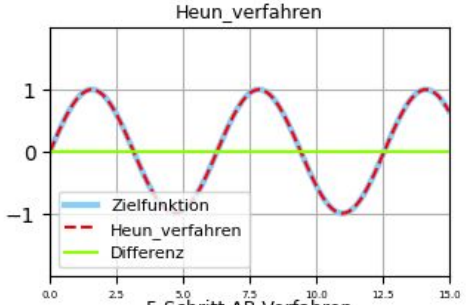
tang_func

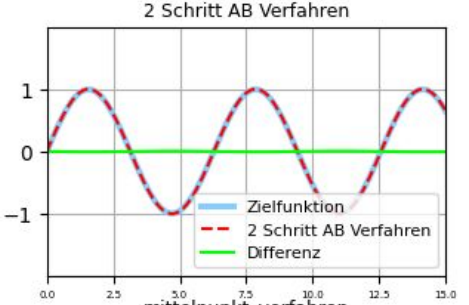

Euler_verfahren
Zielfunktion
Euler_verfahren
Differenz
verbessertes_Euler_verfahren
Zielfunktion
verbessertes_Euler_verfahren
Differenz
Heun_verfahren
Zielfunktion
Heun_verfahren
Differenz
2 Schritt AB Verfahren
Zielfunktion
2 Schritt AB Verfahren
Differenz
3 Schritt AB Verfahren
Zielfunktion
3 Schritt AB Verfahren
Differenz
4 Schritt AB Verfahren
Zielfunktion
4 Schritt AB Verfahren
Differenz
5 Schritt AB Verfahren
Zielfunktion
5 Schritt AB Verfahren
Differenz
mittelpunkt_verfahren
Zielfunktion
mittelpunkt_verfahren
Differenz
lsoda
Zielfunktion
lsoda
Differenz
radau.Radau'
Zielfunktion
radau.Radau'
rk.RK23'
Zielfunktion
rk.RK23'
rk.RK45'
Zielfunktion
rk.RK45'
rk.DOP853'
Zielfunktion
rk.DOP853'
bdf.BDF'
Zielfunktion
bdf.BDF'
lsoda.LSODA'
Zielfunktion
lsoda.LSODA'
Differenzen zur Zielfunktion
0
1
2
3
4
5
6
7
8

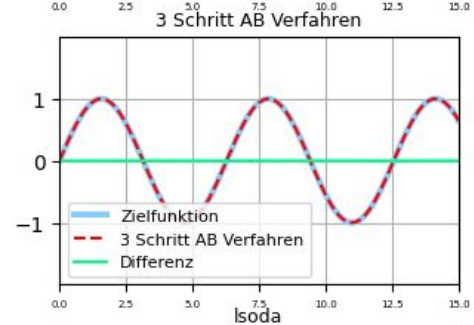
sin_func

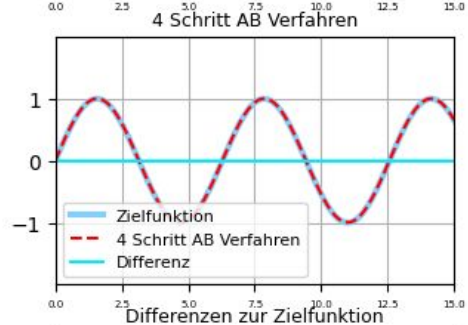

Euler_verfahren
Zielfunktion
Euler_verfahren
Differenz

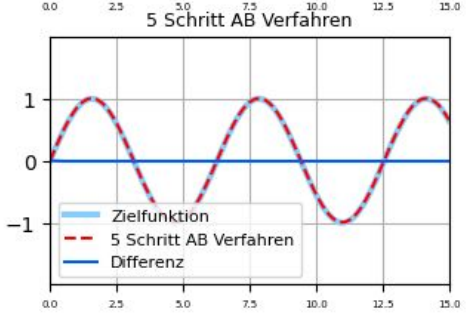

verbessertes_Euler_verfahren
Zielfunktion
verbessertes_Euler_verfahren
Differenz

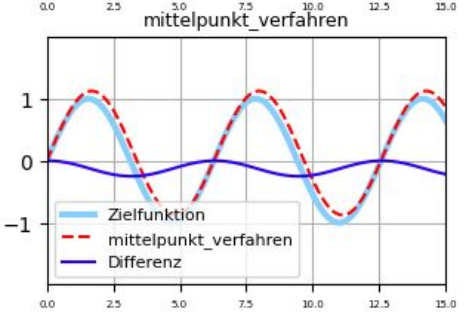

Heun_verfahren
Zielfunktion
Heun_verfahren
Differenz

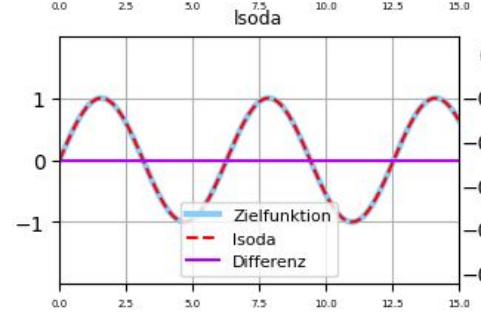

2 Schritt AB Verfahren
Zielfunktion
2 Schritt AB Verfahren
Differenz

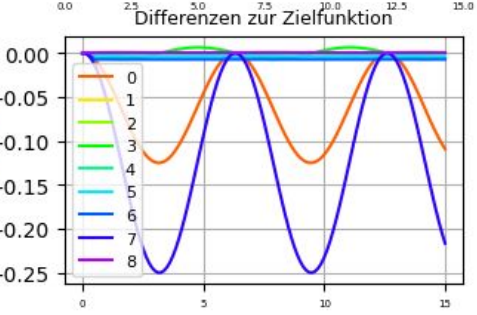

3 Schritt AB Verfahren
Zielfunktion
3 Schritt AB Verfahren
Differenz

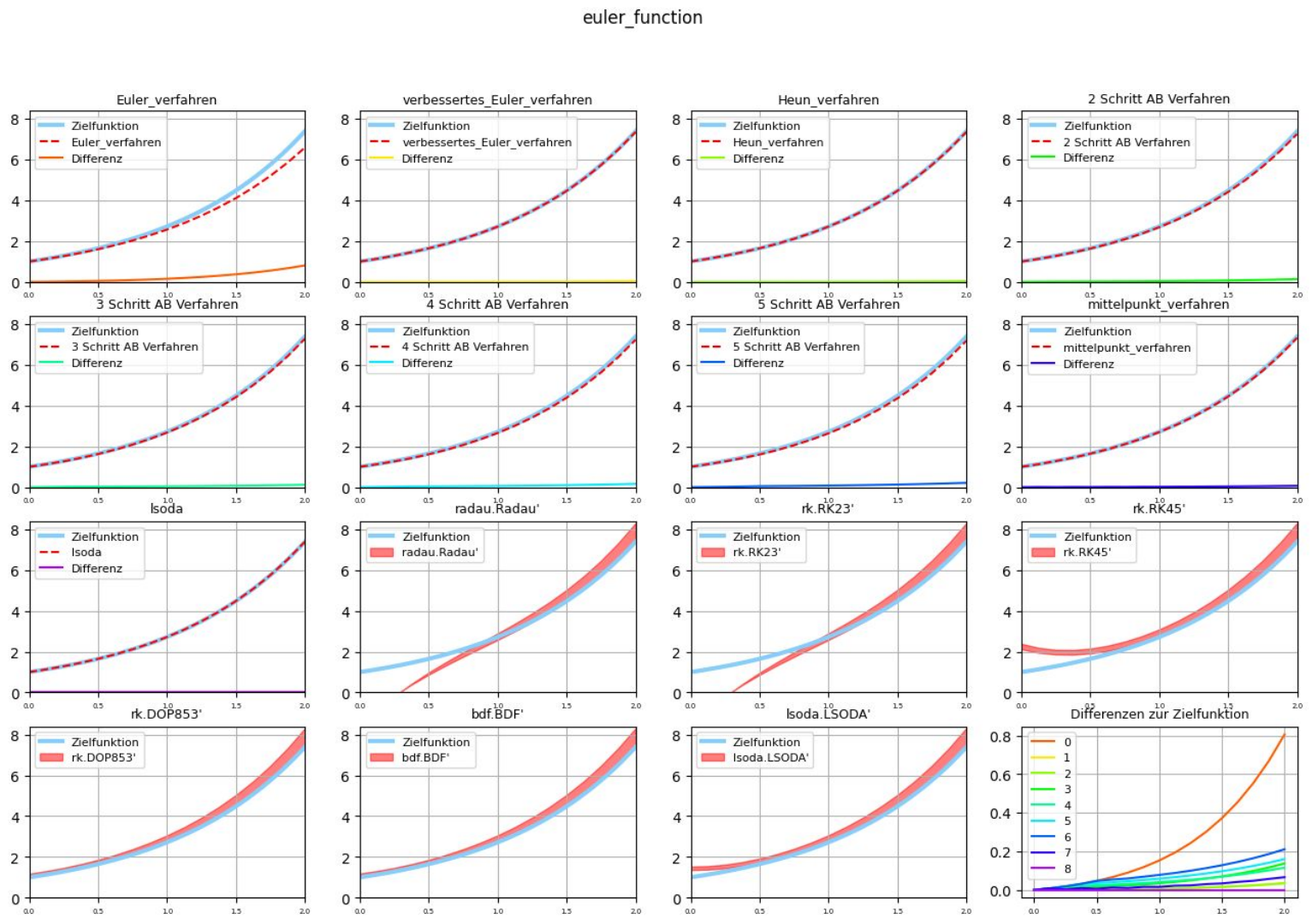

4 Schritt AB Verfahren
Zielfunktion
4 Schritt AB Verfahren
Differenz


5 Schritt AB Verfahren
Zielfunktion
5 Schritt AB Verfahren
Differenz

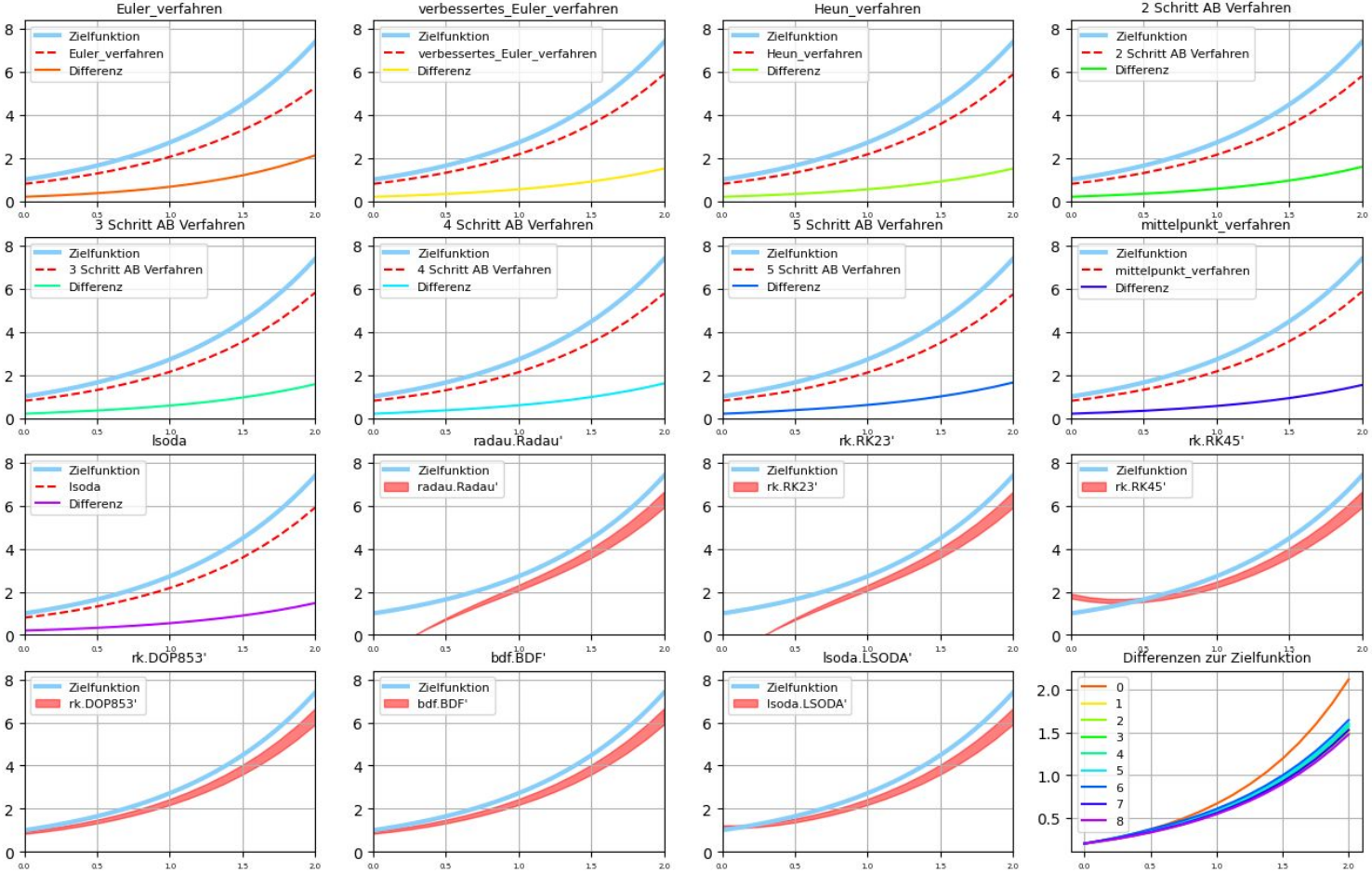

mittelpunkt_verfahren
Zielfunktion
mittelpunkt_verfahren
Differenz


lsoda
Zielfunktion
lsoda
Differenz

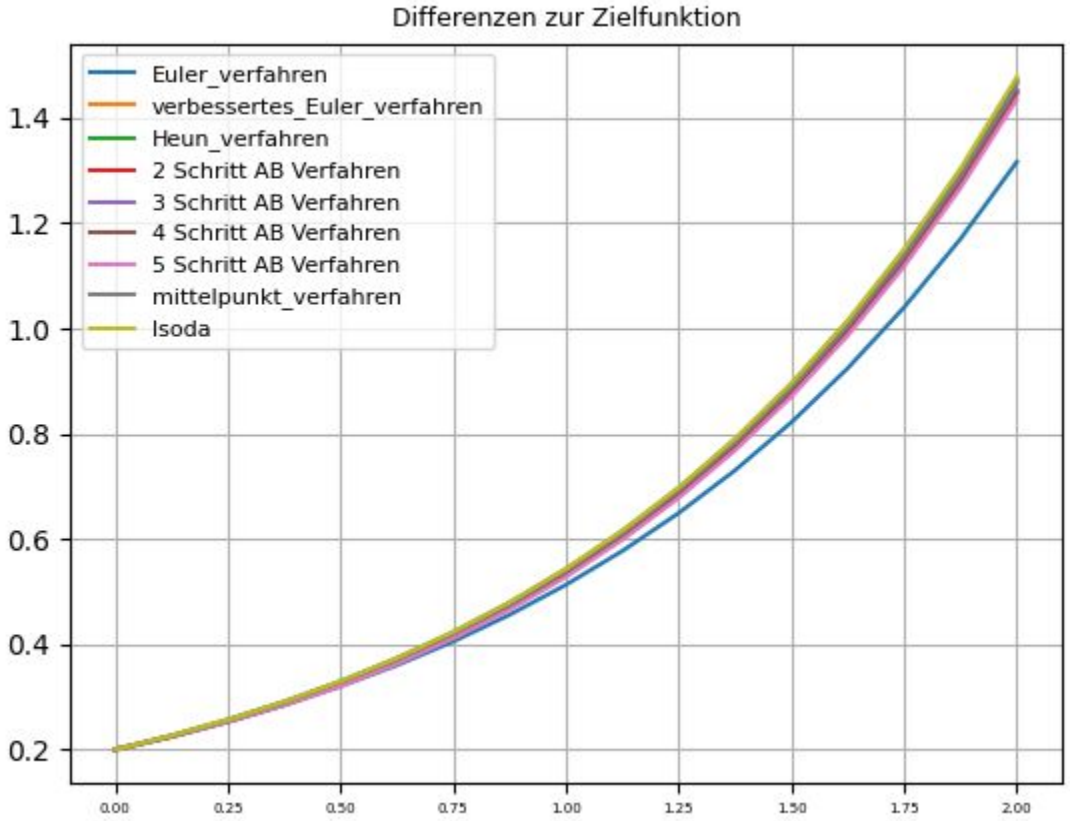

Differenzen zur Zielfunktion
0
1
2
3
4
5
6
7
8


euler_function
Euler_verfahren
Zielfunktion
Euler_verfahren
Differenz
verbessertes_Euler_verfahren
Zielfunktion
verbessertes_Euler_verfahren
Differenz
Heun_verfahren
Zielfunktion
Heun_verfahren
Differenz
2 Schritt AB Verfahren
Zielfunktion
2 Schritt AB Verfahren
Differenz
3 Schritt AB Verfahren
Zielfunktion
3 Schritt AB Verfahren
Differenz
4 Schritt AB Verfahren
Zielfunktion
4 Schritt AB Verfahren
Differenz
5 Schritt AB Verfahren
Zielfunktion
5 Schritt AB Verfahren
Differenz
mittelpunkt_verfahren
Zielfunktion
mittelpunkt_verfahren
Differenz
lsoda
Zielfunktion
lsoda
Differenz
radau.Radau'
Zielfunktion
radau.Radau'
rk.RK23'
Zielfunktion
rk.RK23'
rk.RK45'
Zielfunktion
rk.RK45'
rk.DOP853'
Zielfunktion
rk.DOP853'
bdf.BDF'
Zielfunktion
bdf.BDF'
lsoda.LSODA'
Zielfunktion
lsoda.LSODA'
Differenzen zur Zielfunktion
0
1
2
3
4
5
6
7
8

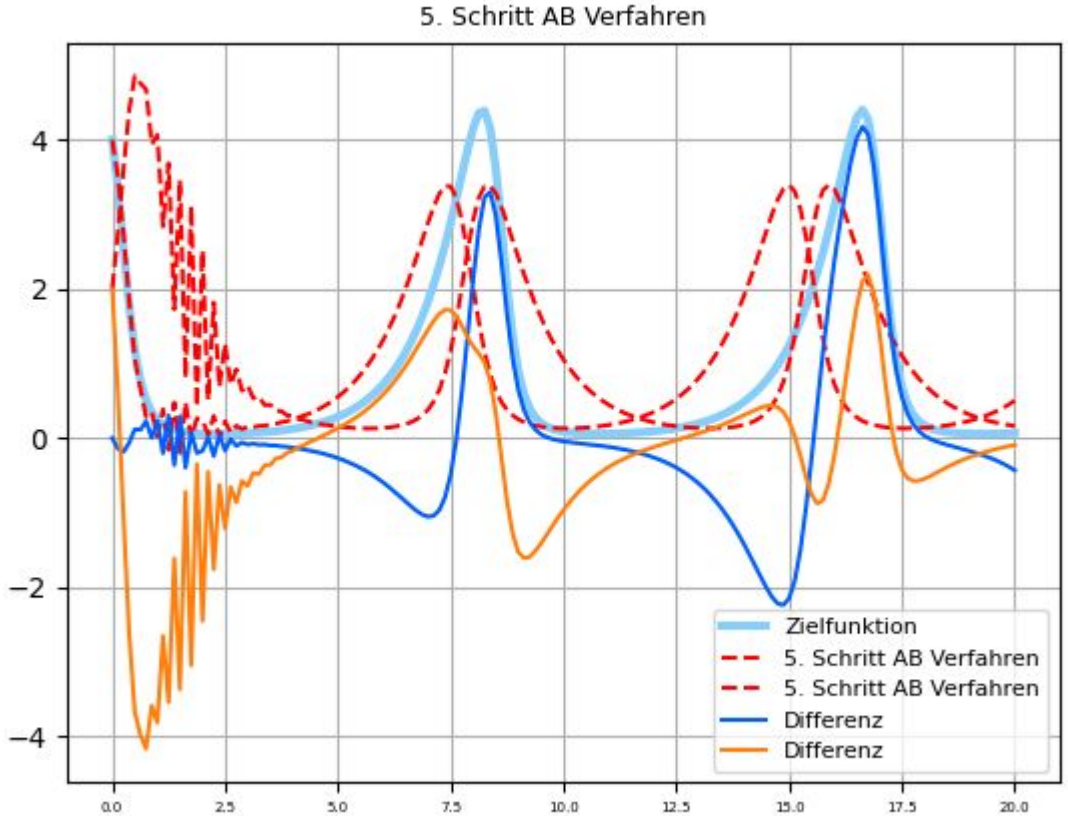
Startwert 0.8



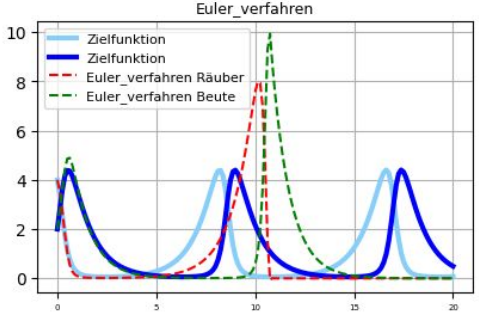
Fehler bei Startwert

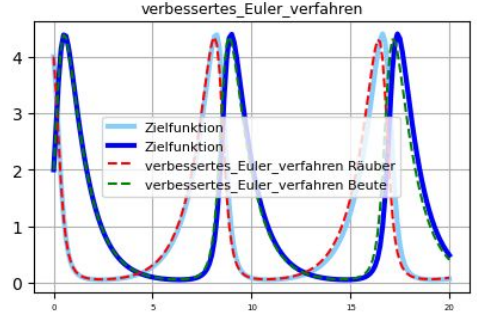


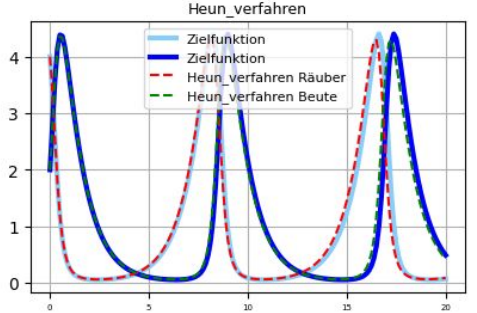
Probleme bei der Darstellung

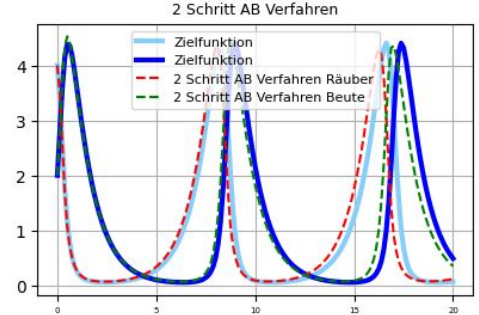


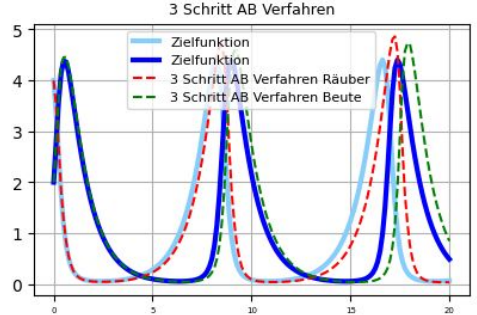
Lotka_Volterra

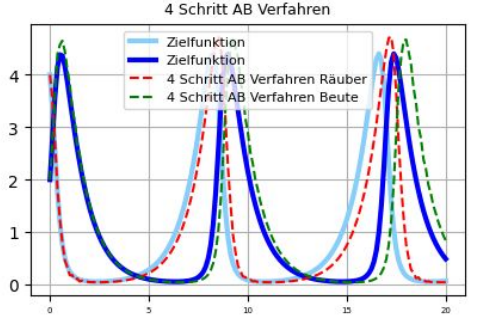


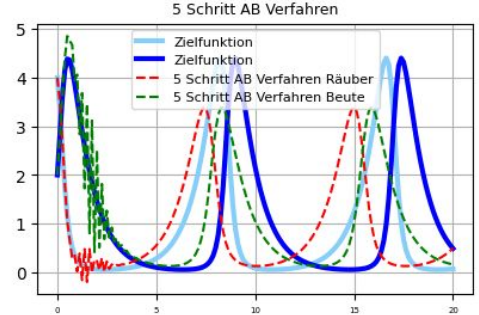


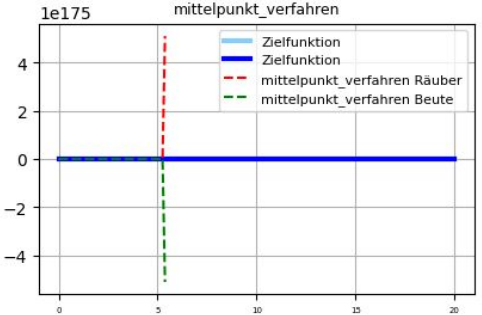












Das Euler-Verfahren hat 16 Funktionsaufrufe über 16 Schritten also eine Rate von 1.0 Aufrufen pro Schritt

Das verbesserte Euler-Verfahren hat 32 Funktionsaufrufe über 16 Schritten also eine Rate von 2.0 Aufrufen pro Schritt

Das Heun-Verfahren hat 48 Funktionsaufrufe über 16 Schritten also eine Rate von 3.0 Aufrufen pro Schritt

Das 2 Schritt AB Verfahren hat 31 Funktionsaufrufe über 16 Schritten also eine Rate von 1.9375 Aufrufen pro Schritt

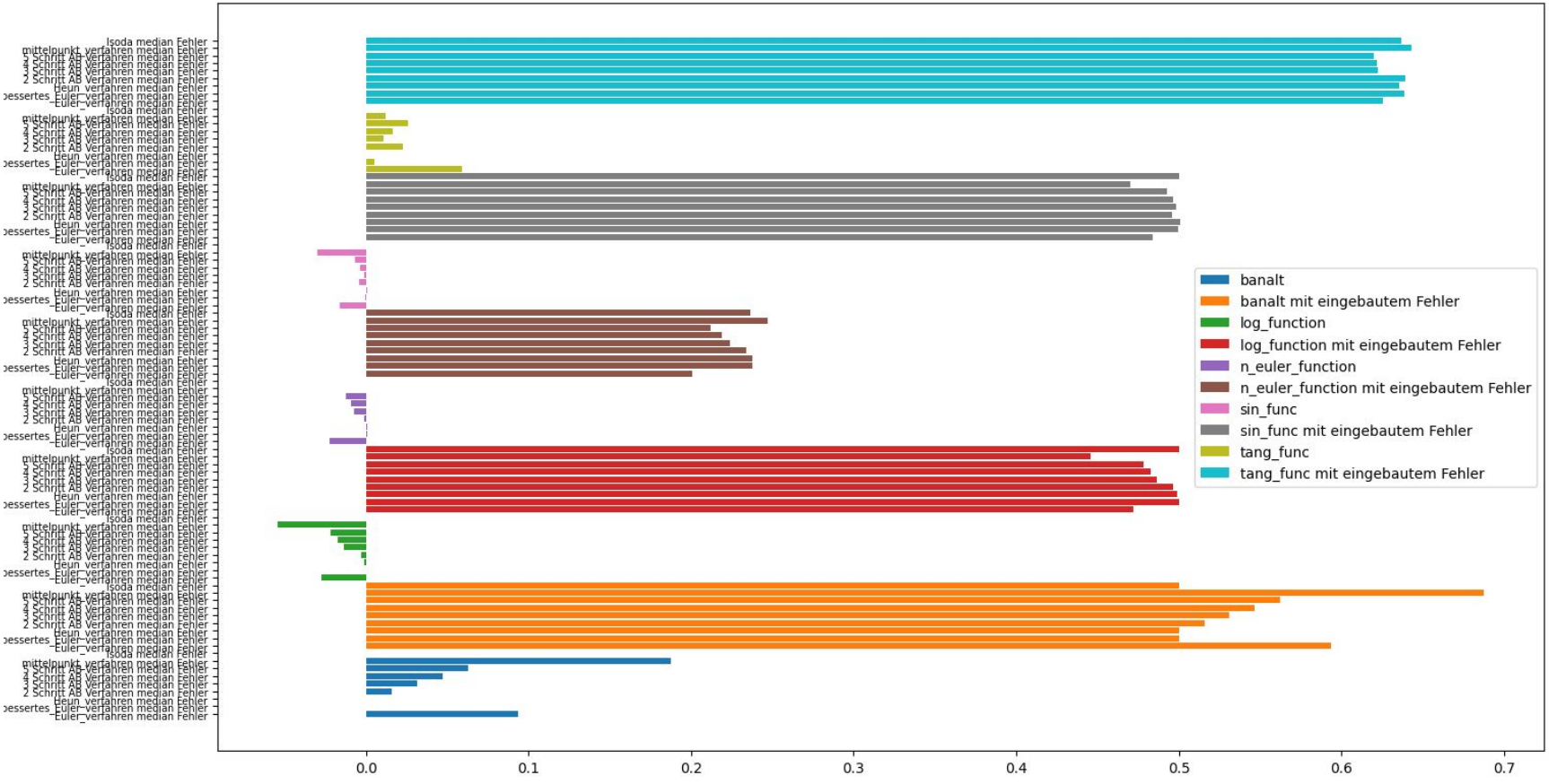
Das 3 Schritt AB Verfahren hat 44 Funktionsaufrufe über 16 Schritten also eine Rate von 2.75 Aufrufen pro Schritt

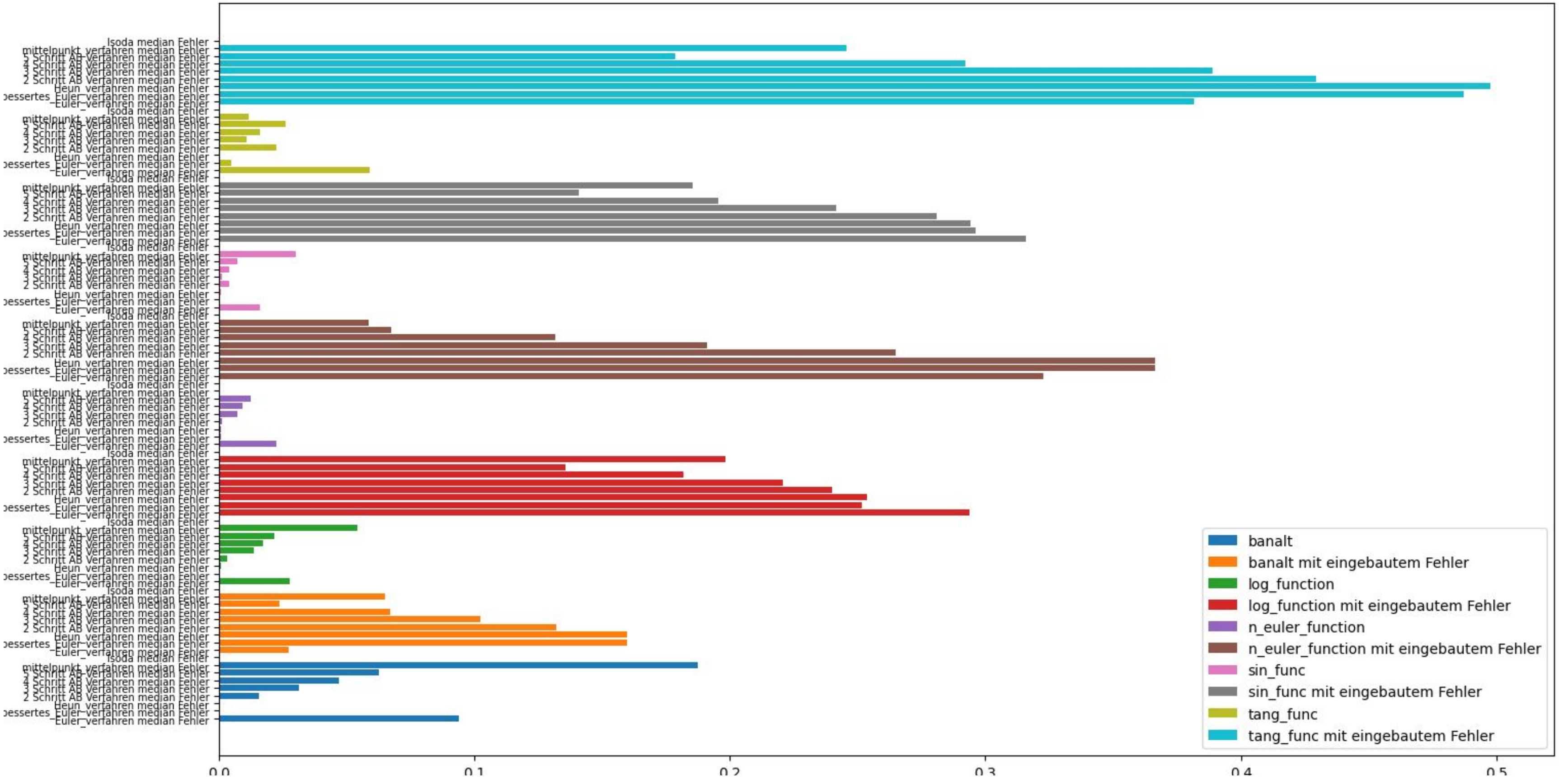
Das 4 Schritt AB Verfahren hat 55 Funktionsaufrufe über 16 Schritten also eine Rate von 3.4375 Aufrufen pro Schritt

Das 5 Schritt AB Verfahren hat 64 Funktionsaufrufe über 16 Schritten also eine Rate von 4.0 Aufrufen pro Schritt

Das Mittelpunkt-Verfahren hat 16 Funktionsaufrufe über 16 Schritten also eine Rate von 1.0 Aufrufen pro Schritt

Median Fehler bei Starwert um -0.5 verschoben





lsoda median Fehler
mittelpunkt_verfahren median Fehler
5 Schritt AB Verfahren median Fehler
4 Schritt AB Verfahren median Fehler
3 Schritt AB Verfahren median Fehler
2 Schritt AB Verfahren median Fehler
Heun_verfahren median Fehler
besssertes_Euler_verfahren median Fehler
Euler_verfahren median Fehler
0.0
0.1
0.2
0.3
0.4
0.5
banalt
banalt mit eingebautem Fehler
log_function
log_function mit eingebautem Fehler
n_euler_function
n_euler_function mit eingebautem Fehler
sin_func
sin_func mit eingebautem Fehler
tang_func
tang_func mit eingebautem Fehler

Literaturverzeichnis

https://www-m2.ma.tum.de/foswiki/pub/M2/Allgemeines/NODE/folien_Runge_Kutta.pdf

https://www-m2.ma.tum.de/foswiki/pub/M2/Allgemeines/NumerikSS12/folien_ode_pt3.pdf

A. Quarteroni, R. Sacco, F. Saleri: Numerische Mathematik 2 - Springer (2002)

https://docs.scipy.org/doc/scipy/reference/integrate.html

https://me-lrt.de/v1-6-mehrschrittverfahren