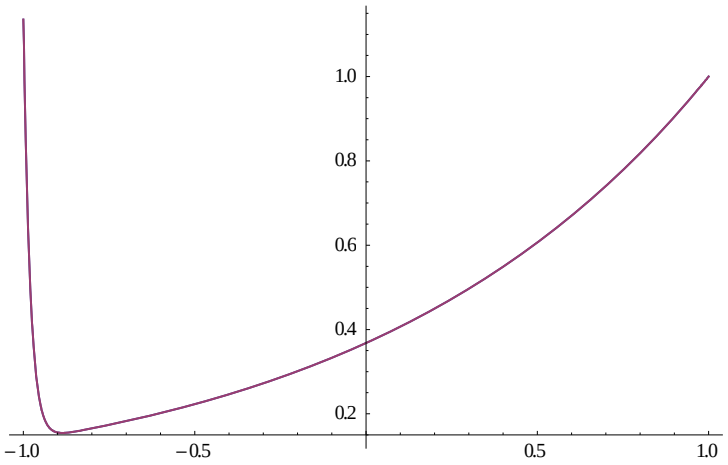
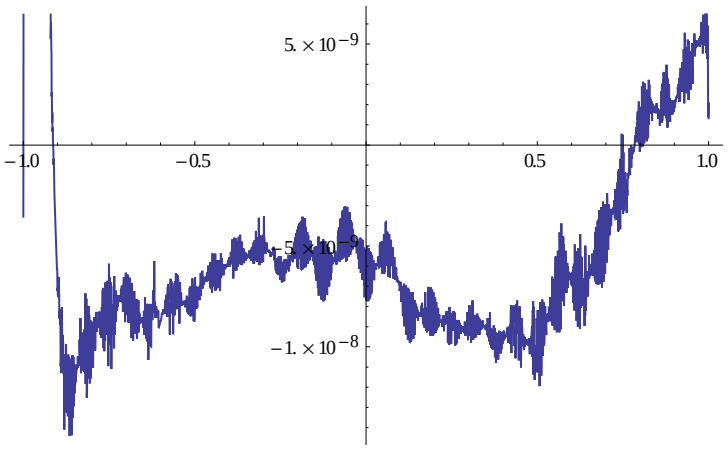
$k_{max}^{(0)}$	j	r	Q_{min}	Q_a
30	3	2	6.93893×10^{-9}	18.6674
30	3	3	8.46827×10^{-6}	1.18102
35	4	4	3.44103×10^{-10}	0.000228873

For critical examples we could start with a higher k_{max}, j and r. The solution of the first step (with $j = 3$ and $r = 2$) is not bad, the big Q_a of 18.6674 results form a big residual value at one point (t = -59/60).

Here are the graphs of y_4 and y, where we see a good approximation, too.



Here is the graph of y_4 - y:



Example III

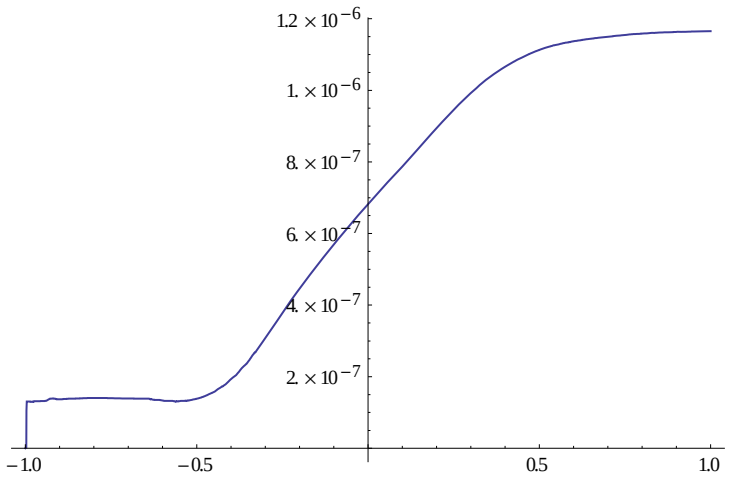
We apply the algorithm on a second order ODE with boundary conditions:

$y'' = (-t \cdot y' - \mu \cdot \pi^2 \cdot \cos(\pi t) - \pi t \sin(\pi t)) / \mu$ with $\mu = 1/10$ and with $y(-1) = -2$, $y(1) = 0$, the approximation interval is $I = [-1, 1]$.

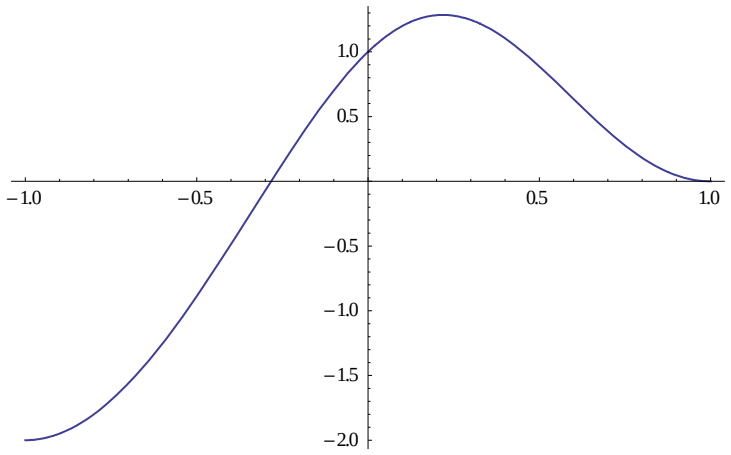
We called the module for the algorithm with:

```
WCollocationS2Alg[π² Cos[π t]+10 π t Sin[π t]+10 t y'[t]+y''[t],{-1,1},{-2,
0},-1.,1.,15,1,2,2]
```

Here is the graph of $\eta - y$ (η is the NDSolve solution):



Here is the graph of η:

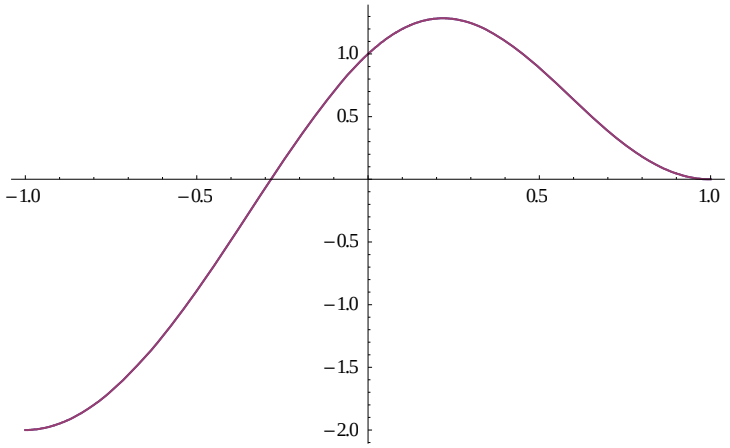


Now we see the iteration-protocol:

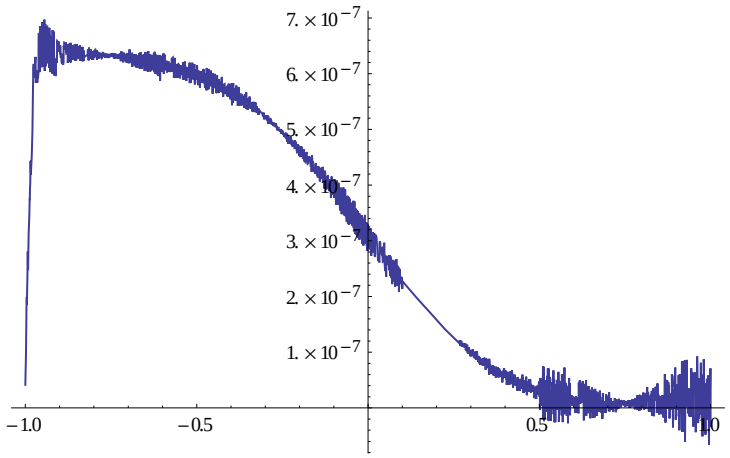
$k_{max}^{(0)}$	j	r	Q_{min}	Q_a
15	1	2	5.8001×10^{-13}	1.85559×10^{-7}

For critical examples we could start with a higher k_{max}, j and r.

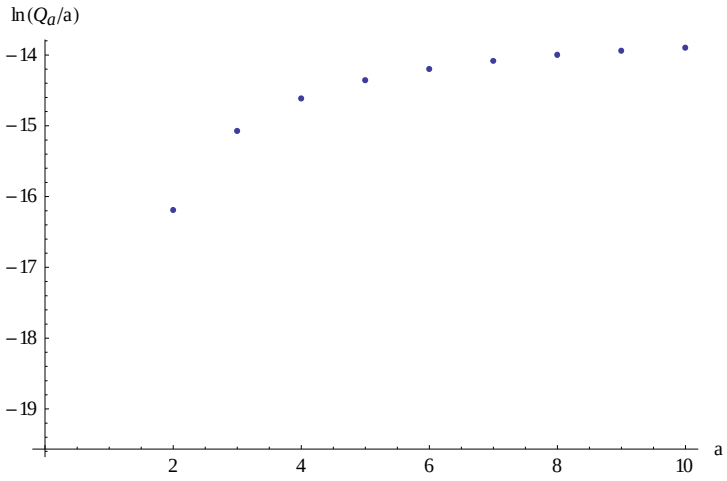
Here are the graphs of y_1 and y (we see no differences):



Here is the graph of y_1 - y:



At last we see graphically the relation between a und Q_a/a in this example:

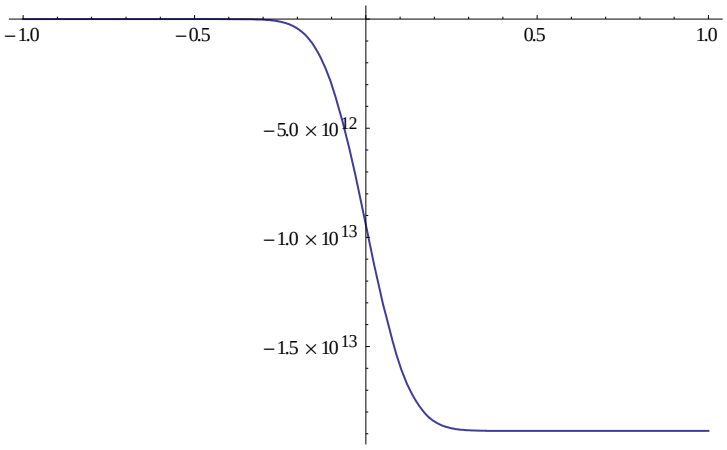


When we set $\mu = 1/100$, then NDSolve get problems:

```
NDSolve::bvluc: The equations derived from the boundary conditions are numerically
ill-conditioned. The boundary conditions may not be sufficient to uniquely define a
solution. The computed solution may match the boundary conditions poorly. >>
```

```
NDSolve::berr: There are significant errors _{0.,-454021.}_ in the boundary value
residuals. Returning the best solution found. ≫
```

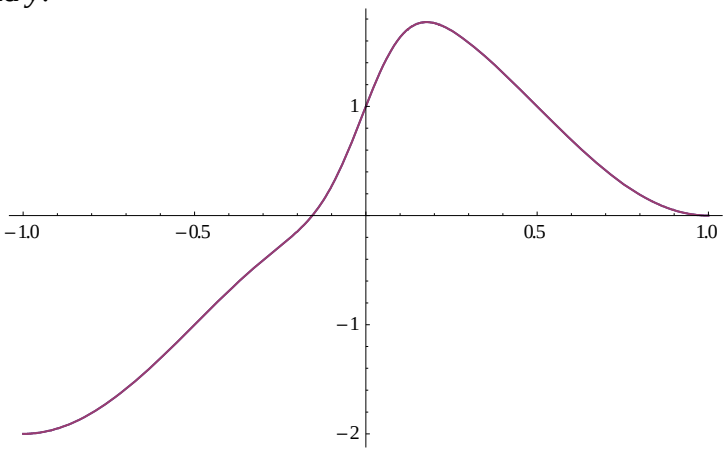
Here we get a very bad approximation:
$\eta - y$ (η is the NDSolve solution):



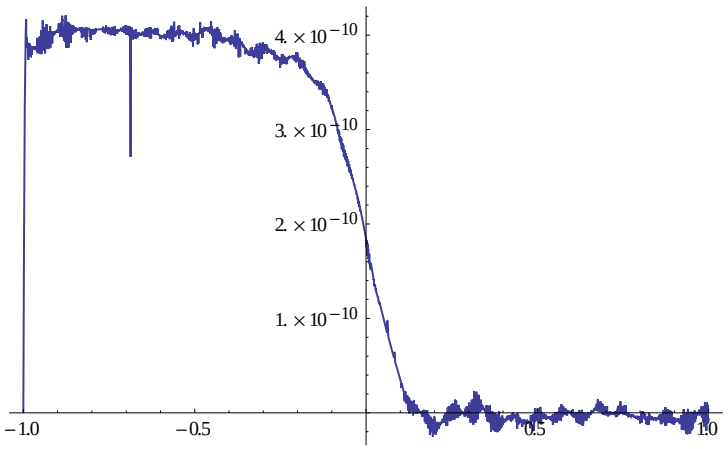
The algorithm has no problems:

```
WCollocationS2Alg[π^2 Cos[π t]+100 π t Sin[π t]+100 t y'[t]+y''[t],{-1,1},{-2,
0},-1.,1.,15,1,2,2]
```

The graphs of y_4 and y:



The graphs of $y_4 - y$:



At last the iteration protocol:

$k_{max}^{(0)}$	j	r	Q_{min}	Q_a
15	1	2	1.44579	319690.
20	1	2	1.97176	258.584
25	2	3	1.66245	195.992
30	3	4	0.00284779	5.50948
35	4	5	4.75163×10^{-14}	1.06944×10^{-10}

Example IV

In this example we will see, that the maximum number of k_{max} in the module (the maximum value of *kmax* was set to 45) should be larger in problems, which needs a large j. We apply the algorithm on a second order ODE with boundary conditions:

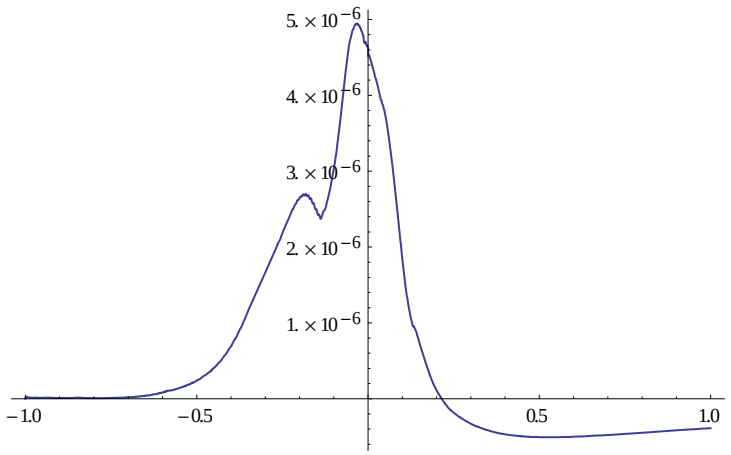
$y'' = (-4t\,y' - 2y)/(\mu + t^2)$ with $\mu = 1/50$ and with $y(-1) = 1/(1 + \mu)$, $y(1) = 1/(1 + \mu)$, the approximation interval is $I = [-1, 1]$.

We called the module for the algorithm with:

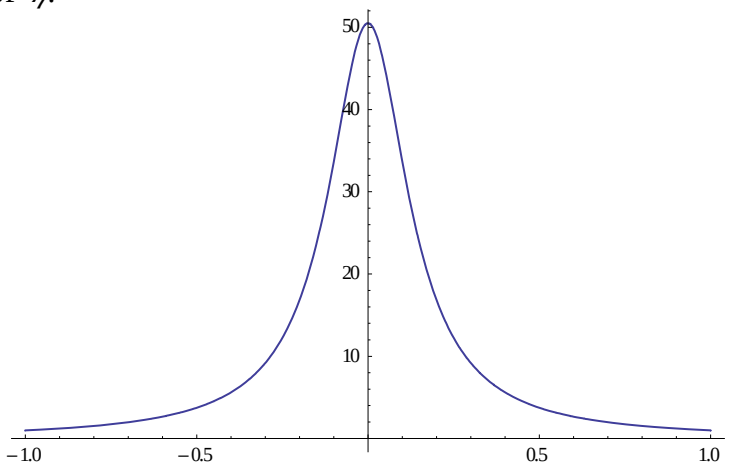
```
WCollocationS2Alg[-(-2y[t]- 4t*y'[t])/(1/50+t^2)+y''[t],{-1,1},{10/11,10/11},
-1.,1.,30,5,3,2]
```

Mathematica NDSolve has no problems:

Here is the graph of $\eta - y$ (η is the NDSolve solution):



Here is the graph of η:



With the used starting values the algorithm stops after the maximum number of steps has been made and the warning came that the solution does not satisfy the convergence criteria $Q_{min} \leq \epsilon_1$ and $Q_a \leq \epsilon_2$. So we know that the solution is not usable.

Now we see the iteration-protocol:

$k_{max}^{(0)}$	j	r	Q_{min}	Q_a
30	5	3	1.96059	1.96059
35	5	3	0.96076	22.8715
40	6	4	1.96059	1.96059
45	7	5	1.96059	1.9606
45	8	6	1.96059	1.96115
45	9	7	1.96059	1.96099
45	10	8	1.96059	2.49262

The module prints: `Warning: Q_min or Q_a is bigger than the tolerance!`

We see in the iteration protocol, that the module has a maximum number of k_{max}. For critical problems, where we need a bigger j, the maximum number of k_{max} should be set to a higher value than 45 in `WCollocationS2Alg`. With $k_{max}^{(0)}$ less than 2^j the method cannot get a solution (with a small Q_{min}, because Q_{min} is in that case $\geq y(0)^2 + y(1)^2$) with the Shannon ϕ with that boundary conditions, because at $t_{end} = 1$ we get the boundary condition

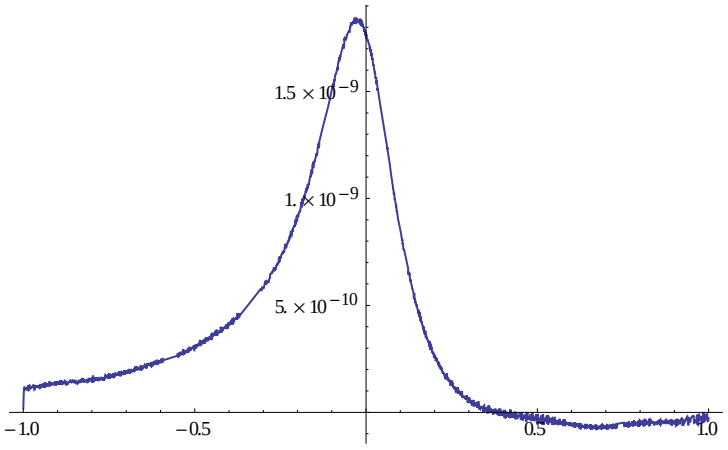
$$y_j(1) := \sum_{k=k_{min}}^{k_{max}} c_k \cdot \phi_{j,k}(1) = \sum_{k=k_{min}}^{k_{max}} c_k \cdot 2^{j/2} \phi(2^j \cdot 1 - k) \overset{!}{=} y(1)$$

and $\phi(m) = 0$ for integer $m \neq 0$ and $\phi(1) = 1$. So if k_{max} is less than 2^j the boundary condition cannot be fulfilled if $y(1) \neq 0$. Because if $y(-1) \neq 0$ we get the same for k_{min}. So with that boundary conditions we get $k_{min} \leq -2^j$ and $k_{max} \geq 2^j$. Otherwise $y_j(\pm 1) = 0$. With integer values of the boundaries t_0 and t_{end} general k_{max} should be greater or equal $2^j t_{end}$. Because of k_{min} should be less or equal $2^j t_0$, in the module $k_{max}^{(0)}$ should be greater or equal (only if the expression is integer) $(2^j t_{end} - 2^j t_0)/2$, because in the module $k_{max}^{(0)}$ is positive (the module shifts automatically the summation area, $k_{max} = k_{max}^{(0)} + k_0$ and $k_{min} = -k_{max}^{(0)} + k_0$).

When set *kmaxmax* = 100 an apply the method with

```
WCollocationS2Alg[-(-2y[t]- 4t*y'[t])/(1/50+t^2)+y''[t],{-1,1},{10/11,10/11},
-1.,1.,80,6,8,2]
```

then the algorithm stops directly with $k_{max} = 80$, $j = 6$, $r = 8$, $Q_{min} = 1.88584 \times 10^{-12}$ and $Q_a = 9.84522 \times 10^{-10}$. Here are the graphs of $y_6 - y$:

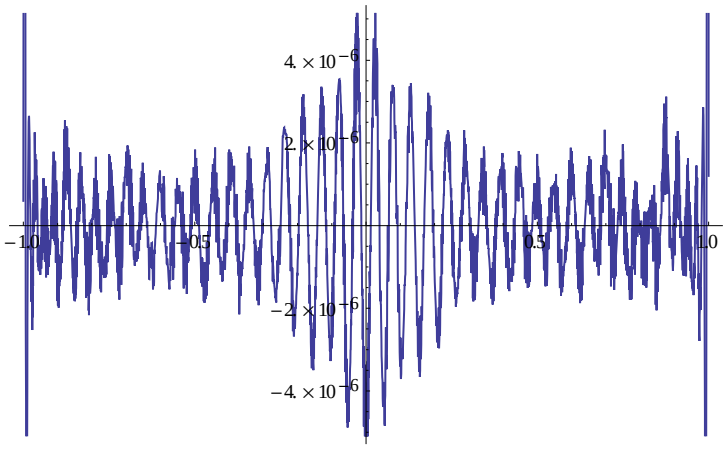


A direct approximation makes no problems, too. For example, with

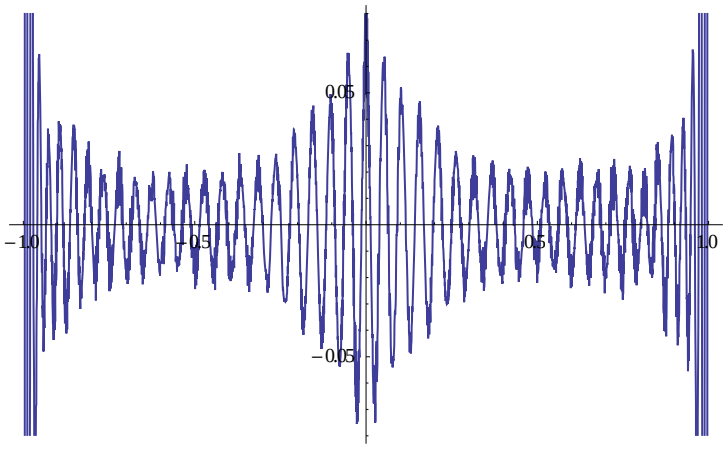
```
WCollocationS2Alg[y[t]-fe[t],-1,50/51,-1.,1.,25,2,4,2]
Fe[t_]:=50/(1+50t^2)
```

the algorithm stops with k_{max} = 45, j = 5, r = 7, Q_{min} = 5.83423×10^{-10} and Q_a = 4.52406×10^{-9} .

Here are the graphs of y_5 - y:



The difference between the second derivations is relative large (graph of $y_5'' - y''$):



Here we get:

$$\sum_{i=0}^{m}(y_j(t_i)-y(t_i))^2 = 5.83423\times10^{-10}$$

with $h=\frac{t_{end}-t_0}{m}$ with $m = r\cdot|k_{max}^{(0)}| = 45\cdot7$, t_0 = -1 and t_{end} = 1.

The derivation of the second order derivatives is much larger:

$$\sum_{i=1}^{m}(y_j''(t_i)-y''(t_i))^2 = 8308.59$$

The biggest difference we get at the beginning and at the end of the approximation interval with 3970.8 and 4307.34.

Comparing the L^2 approximation with the direct approximation on the interval [-1,1]:
We can not apply the information about the $L^2(R)$ approximation $\tilde{y}_j$ from y on

$$\underbrace{span\{\phi_{j,k}\}_{k=k_{min},k_{min}+1,\ldots,k_{max}}}_{=:S_j} \subset V_j$$

to get the right k_{min} and k_{max} for the algorithm, because the $L^2(R)$ approximation may need a lot bigger k_{max} than the direct approximation through (4) on the interval [-1, 1] (to get nearly the same quality of approximation), like in our example, where the decay of the coefficients $\tilde{c}_k$ is very poor.

For the $L^2(R)$ approximation the coefficients will be calculated as usual with orthogonal bases (we assume for easier notation, that the scaling function and y is real valued):

$$\tilde{c}_k = \langle y, \phi_{j,k} \rangle_{L^2(R)} = \int_R y(t) \cdot \phi_{j,k}(t) dt$$

And so we get the orthogonal projection from y on S_j:

$$\tilde{y}_j(t) := \sum_{k=k_{min}}^{k_{max}} \tilde{c}_k \cdot \phi_{j,k}(t)$$

Generally $\|\tilde{y}_j - y\|_{L^2(I)} \geq \|\hat{y}_j - y\|_{L^2(I)}$ with $I \subset R$. Here $\tilde{y}_j$ is the best approximation on R, calculated through

$$min \|y_j - y\|_{L^2(R)} = \|\tilde{y}_j - y\|_{L^2(R)}$$

and $\hat{y}_j$ is the best approximation on I, calculated through

$$(3) \quad min \|y_j - y\|_{L^2(I)} = \|\hat{y}_j - y\|_{L^2(I)} .$$

The reason for that is because $\tilde{y}_j$ is the best approximation according to the $L^2(R)$ norm on R but $\hat{y}_j$ is the best approximation from y only on the interval I as a part of R. The direct approximation is the numerical solution of the minimum problem (3) above, so y_j is the numerical approximation of $\hat{y}_j$ or the solution of (4). Here - for easier notation - we named the solution of the minimum problems the same as the unknown functions.

Theoretically we would get the solution of the continuous minimum problem (3) through the following considerations:

We calculate instead of (3) the orthogonal projection of a function $\breve{y}$ on S_j. The function $\breve{y}$ is on I identical to y and on $R \setminus I$ identically to our function y_j of S_j as a part of V_j. So $\breve{y} = y_I + y_{R\setminus I}$ (where y_I vanishes on R\I and $y_{R\setminus I}$ vanishes on I):

$$\breve{y}(t) = 1_I(t) \cdot y(t) + 1_{R\setminus I}(t) \cdot \sum_{k=k_{min}}^{k_{max}} c_k \cdot \phi_{j,k}(t) \text{ with indicator function 1.}$$

So we approximate y only on I .Outside I the approximation function has no restricts. An other and a worse approximation we would get through the orthogonal projection from $1_I \cdot y$ on V_j. The reason is that we would cut the function y and this would lead generally to a bad decay behavior in the Fourier space, see [19].

Here we get the coefficients c_k through:

$$c_k = \left\langle \breve{y}, \phi_{j,k} \right\rangle_{L^2(R)} = \left\langle y, \phi_{j,k} \right\rangle_{L^2(I)} + \left\langle y_j, \phi_{j,k} \right\rangle_{L^2(R\setminus I)} = \left\langle y, \phi_{j,k} \right\rangle_{L^2(I)} + \sum_{k=k_{min}}^{k_{max}} c_l \cdot \left\langle \phi_{l,k}, \phi_{j,k} \right\rangle_{L^2(R\setminus I)}$$

For I = R we would get the best approximation on R through $\left\langle y, \phi_{j,k} \right\rangle_{L^2(R)}$. $\{\phi_{j,k}\}_k$ is general no orthogonal system on $L^2(R \setminus I)$ (only if the support of $\phi_{j,k}$ is in $R \setminus I$). So we get:

$$c_k - \sum_{k=k_{min}}^{k_{max}} c_l \cdot \left\langle \phi_{l,k}, \phi_{j,k} \right\rangle_{L^2(R\setminus I)} = \left\langle y, \phi_{j,k} \right\rangle_{L^2(I)}$$

$$\sum_{k=k_{min}}^{k_{max}} c_l \cdot \delta_{l,k} - c_l \cdot \left\langle \phi_{l,k}, \phi_{j,k} \right\rangle_{L^2(R\setminus I)} = \left\langle y, \phi_{j,k} \right\rangle_{L^2(I)}$$

$$\sum_{k=k_{min}}^{k_{max}} c_l \cdot \left\langle \phi_{l,k}, \phi_{j,k} \right\rangle_{L^2(R)} - c_l \cdot \left\langle \phi_{l,k}, \phi_{j,k} \right\rangle_{L^2(R\setminus I)} = \left\langle y, \phi_{j,k} \right\rangle_{L^2(I)}$$

$$\sum_{k=k_{min}}^{k_{max}} c_l \cdot \underbrace{\left\langle \phi_{l,k}, \phi_{j,k} \right\rangle_{L^2(I)}}_{:=a_{l,k}} = \underbrace{\left\langle y, \phi_{j,k} \right\rangle_{L^2(I)}}_{:=u_k} \quad \text{, for } l = k_{min}, \ldots, k_{max} \qquad (5)$$

That is the normal equation for the vector c: $Ac = u$ and if we calculate c through this equation we get the approximation error of (3):

$$min \left\| y_j - y \right\|_{L^2(I)} = \left\| \hat{y}_j - y \right\|_{L^2(I)} = \sqrt{\left(\left\| y \right\|_{L^2(I)} \right)^2 - c^T A c} \qquad (6)$$

The approximation error of the global $\hat{y}_j$ approximation on I is:

$$\left\| \tilde{y}_j - y \right\|_{L^2(I)} = \sqrt{\left(\left\| y \right\|_{L^2(I)} \right)^2 + \tilde{c}^T A \tilde{c} - 2\tilde{c}^T u} \quad \text{with } \tilde{c}_k = \left\langle y, \phi_{j,k} \right\rangle_{L^2(R)}$$

(6) we get through (which is general for every vector c right):

$$\left(\left\|y_j - y\right\|_{L^2(I)}\right)^2 = \left(\left\|y\right\|_{L^2(I)}\right)^2 - 2\left\langle y, y_j\right\rangle_{L^2(I)} + \left(\left\|y_j\right\|_{L^2(I)}\right)^2$$

with $\left\langle y, y_j\right\rangle_{L^2(I)} = \left\langle y, \sum_{l=k_{min}}^{k_{max}} c_k \cdot \phi_{j,l}\right\rangle_{L^2(I)} = \sum_{l=k_{min}}^{k_{max}} c_k \cdot \left\langle y, \phi_{j,l}\right\rangle_{L^2(I)} = c^T \cdot u$ and

$$\left(\left\|y_j\right\|_{L^2(I)}\right)^2 = \left\langle \sum_{l=k_{min}}^{k_{max}} c_l \cdot \phi_{j,k}, \sum_{k=k_{min}}^{k_{max}} c_k \cdot \phi_{j,k}\right\rangle_{L^2(I)} = \sum_{l=k_{min}}^{k_{max}} \sum_{k=k_{min}}^{k_{max}} c_l \cdot c_k \cdot \left\langle \phi_{j,l}, \phi_{j,k}\right\rangle_{L^2(I)} = c^T Ac$$

So:

$$\left(\left\|y_j - y\right\|_{L^2(I)}\right)^2 = \left(\left\|y\right\|_{L^2(I)}\right)^2 - 2c^T u + c^T Ac$$

For the approximation $\hat{y}_j$ (where we get c through $Ac = u$) we get equation (6).

With a special scalar product $\left\langle x, y\right\rangle_A := x^T Ax$ and the induced norm (with positive definite A) $\left\|x\right\|_A = \sqrt{\left\langle x, y\right\rangle_A}$ we get:

$$\left(\left\|\tilde{y}_j - y\right\|_{L^2(I)}\right)^2 - \left(\left\|\hat{y}_j - y\right\|_{L^2(I)}\right)^2 = c^T Ac + \tilde{c}^T A\tilde{c} - 2\tilde{c}^T u = c^T Ac + \tilde{c}^T A\tilde{c} - 2\tilde{c}^T Ac$$
$$= \left\langle c, c\right\rangle_A + \left\langle \tilde{c}, \tilde{c}\right\rangle_A - 2\left\langle c, \tilde{c}\right\rangle_A = \left\|\tilde{c} - c\right\|_A^2$$

For example:
The orthogonal projection of y on S_6 or V_6 is very near to y, because the differences of the $L^2(R)$ norm the $||Y|| - ||Y_j||$ is very small.

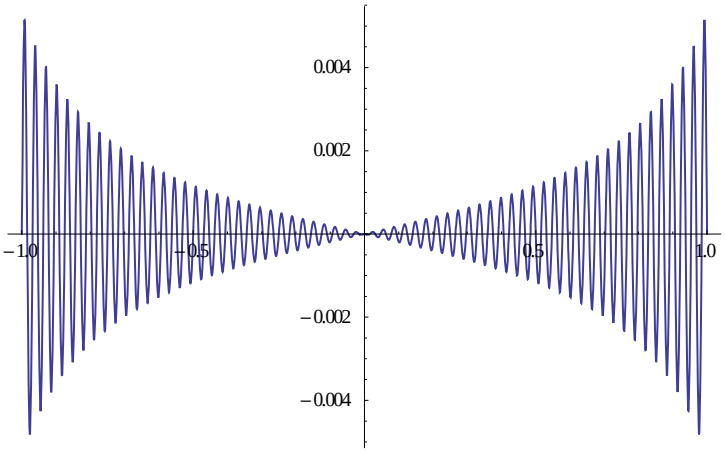
With the direct approximation through

(4) $$min \quad Q(c) = \sum_{i=0}^{m} (y_j'(t_i) - y(t_i))^2$$

We set $k_{max} = 80$ and $j = 6$ ($r = 8$) and start the minimization:

```
fe[t_]:=  50/(1+50t^2)
WCollocationS2Alg[y[t]-fe[t],-1,fe[-1]//N,-1.,1.,80,6,8,2]
```

Here the algorithm stops directly with $k_{max} = 80$, $j = 6$, $r = 8$ and a very small $Q_{min} = 6.98752\times10^{-22}$ and $Q_2 = 4.99871\times10^{-21}$. We started with bigger parameters to get less steps. The difference of the orthogonal projection $\tilde{y}_j$ from y on V_j and y (with $k_{max} = 80$) has the following graph:



Here we get:

$$\sum_{i=0}^{m}(\tilde{y}_6(t_i)-y(t_i))^2 \;=\; 0.00131592 \;\text{ with }\; h=\frac{t_{end}-t_0}{m} \quad \text{with } m = r\cdot|k_{max}^{(0)}| \;= 80\cdot 8.$$

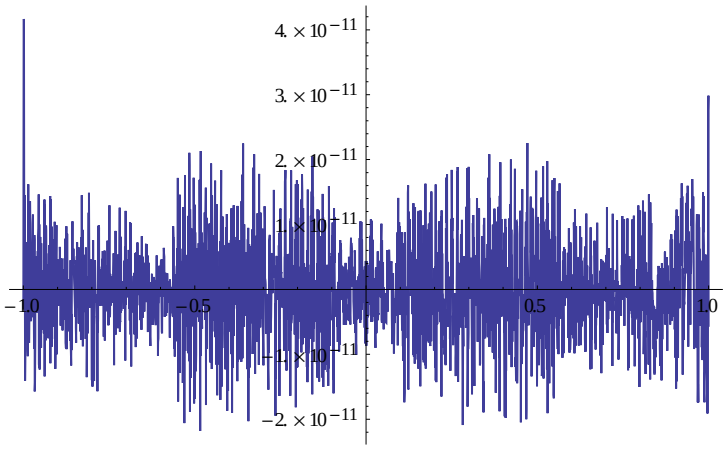
If we set k_{max} = 1000 we get with the same t_i the following sum of squares

$$\sum_{i=0}^{m}(\tilde{y}_6(t_i)-y(t_i))^2 \;=\; 7.42953\times10^{-13}$$

which is small but even much bigger than the sum of squares with k_{max} = 80 and the direct approximation y_6:

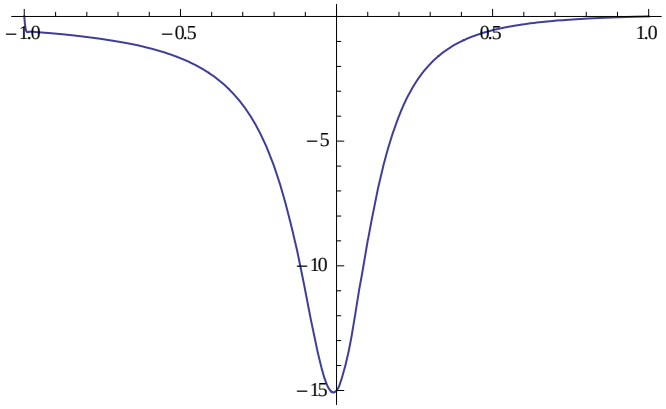
$$\sum_{i=0}^{m}(y_6(t_i)-y(t_i))^2 \;= 6.98752\times10^{-22}$$

Here are the graphs of $y_6 - y$ (k_{max} = 80):



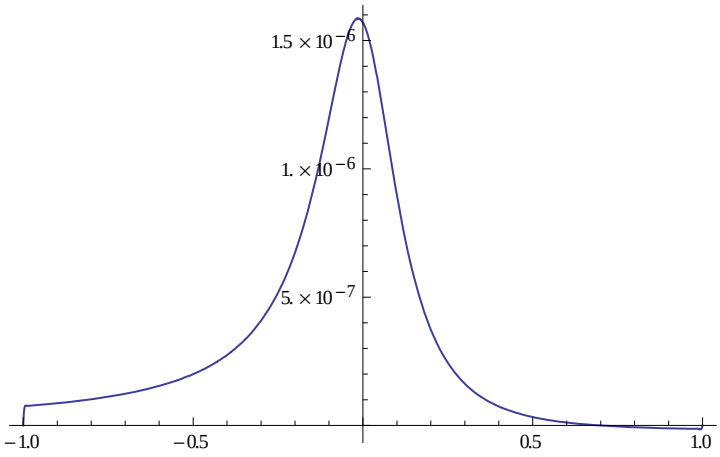
Finally here are some graphs of $y_j - y$ for selected combinations of j, k_{max} and r. Here we can see, that not only Q_{min} but Q_2 must be small, too,. That is what the algorithm does. With a too small r we get a big Q_2. In some cases Q_2 can be large because of big deviations at the edge of the approximation interval. Q_2 was theoretically studied in [16]. In that example we got in many simulations the first good approximations for a minimal value for j of 6 and for k_{max} the minimal value have been 70.

For j = 6, k_{max} = 72, and r = 2 we got a $Q_{min} = 1.82438\times10^{-13}$ and $Q_2 = 9.04114\times10^{7}$:

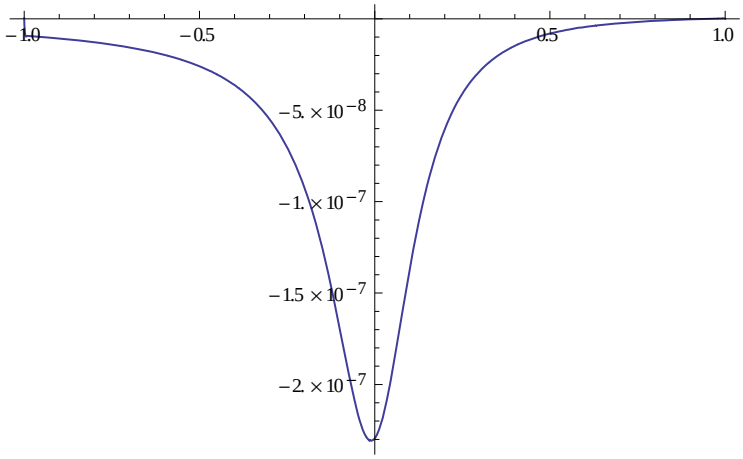


Here Q_2 was too big and so we got a bad approximation. The following examples have decreasing values of Q_2 and the approximation will get successively better.

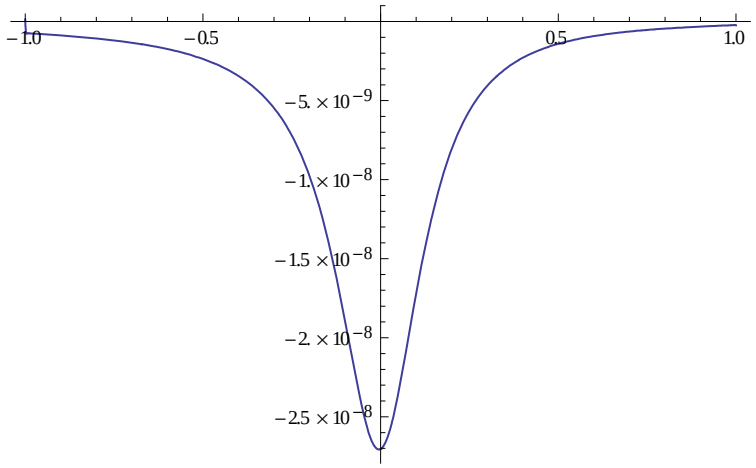
For $j = 6$, $k_{max} = 72$, and $r = 3$ we got a $Q_{min} = 3.31586\times10^{-12}$ and $Q_2 = 5.98751\times10^{-6}$:



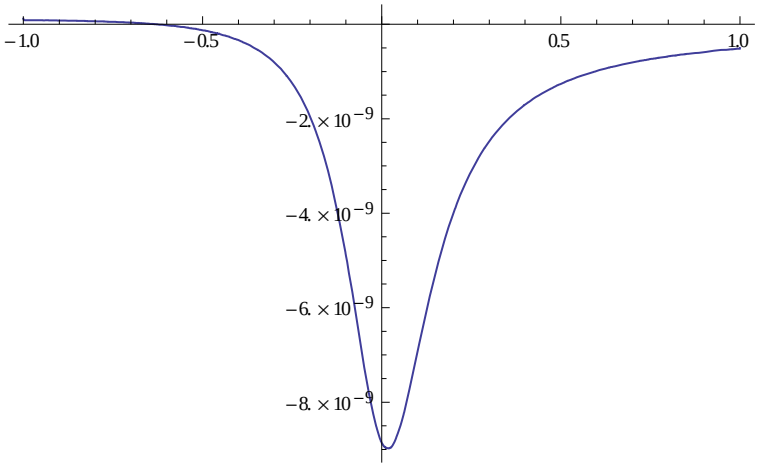
For $j = 6$, $k_{max} = 74$, and $r = 3$ we got a $Q_{min} = 3.35466\times10^{-12}$ and $Q_2 = 3.62539\times10^{-7}$:



For $j = 6$, $k_{max} = 74$, and $r = 5$ we got a $Q_{min} = 4.74696\times10^{-12}$and $Q_2 = 7.57142\times10^{-9}$:



For $j = 6$, $k_{max} = 74$, and $r = 8$ we got a $Q_{min} = 7.71771 \times 10^{-12}$ and $Q_2 = 5.44366 \times 10^{-11}$:



Here we can see how with decreasing values of Q_2 and with Q_{min} in the same magnitude the approximation error decreases.

Example V

We apply the algorithm on a second order ODE with boundary conditions:

$y'' = (y - (\mu \cdot \pi^2 + 1) \cdot \cos(\pi t)) / \mu$ with $\mu = 1/100$ and with $y(-1) = -1$, $y(1) = -1$, the approximation interval is $I = [-1, 1]$.

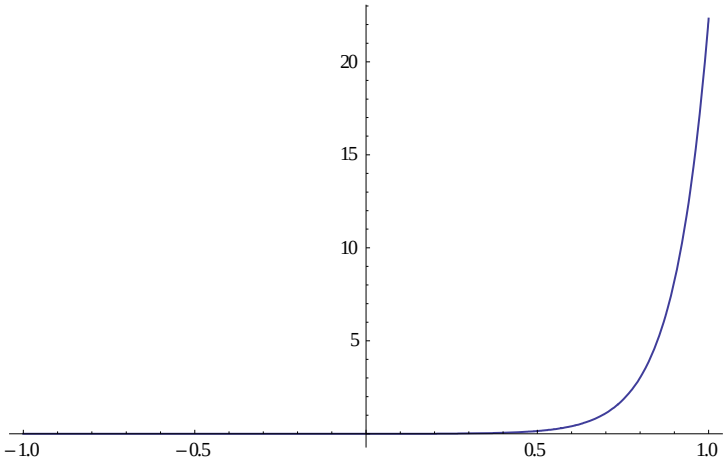
We called the module for the algorithm with:

```
WCollocationS2Alg[-100(-(1+π^2/100) Cos[π t]+y[t])+y''[t],{-1,1},{-1,-1},
-1.,1.,15,1,2,2]
```

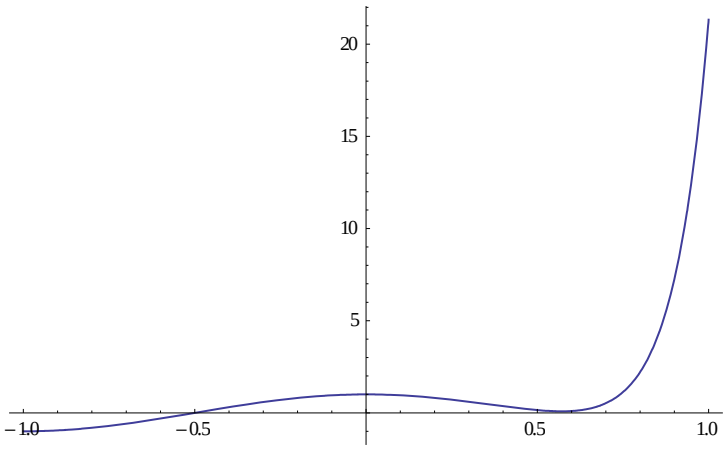
Mathematica NDSolve has problems:

```
NDSolve::bvluc :
The equations derived from the boundary conditions are numerically ill-conditioned.
The boundary conditions may not be sufficient to uniquely define a solution. The
computed solution may match the boundary conditions poorly.
NDSolve::berr: There are significant errors _{0.,-1.76873×10^-7}_ in the boundary
value residuals. Returning the best solution found. ≫
```

Here is the graph of $\eta - y$ (η is the NDSolve solution), were we can see, that the numerical solution of NDSolve has big deviations:



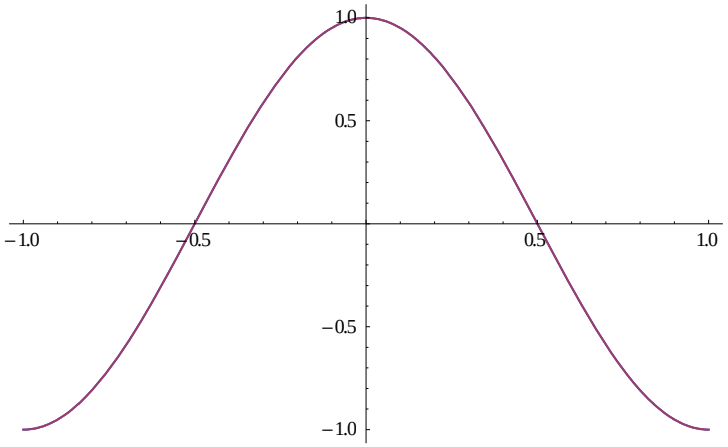
Here is the graph of η:



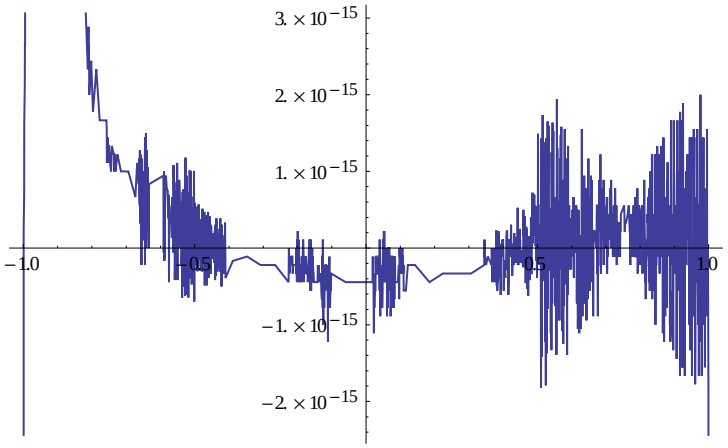
Now we see the iteration-protocol:

$k_{max}^{(0)}$	j	r	Q_{min}	Q_a
15	1	2	6.092×10^{-27}	1.08892×10^{-22}

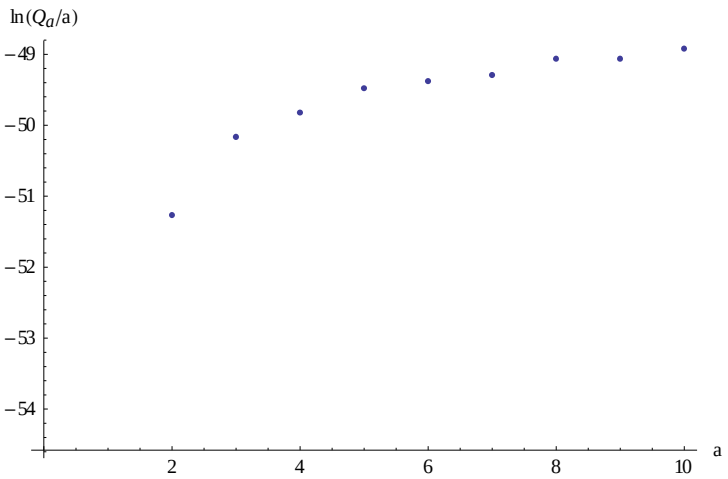
Here are the graphs of y_1 and y (we see no differences):



Here is the graph of y_1 - y:



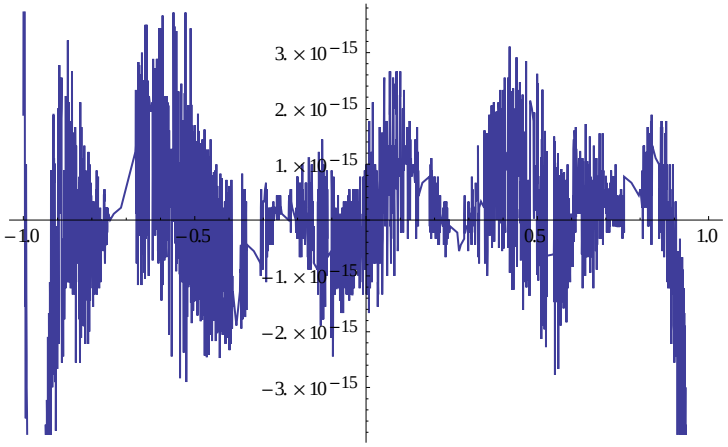
Now we see graphically the relation between a und Q_a/a in this example:



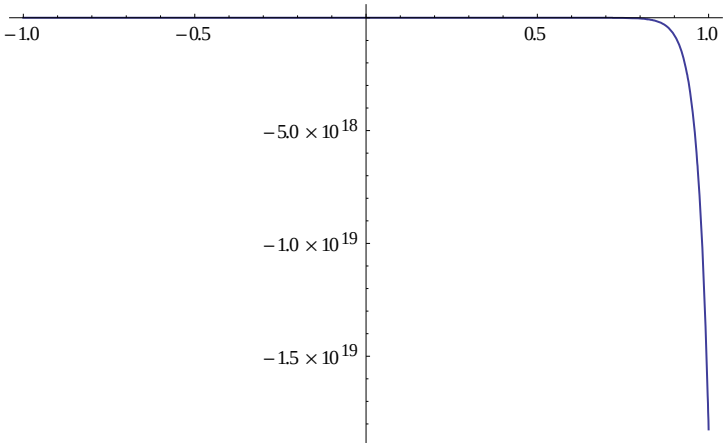
Even with $\mu = 1/1000$ we get after one step a very good approximation:

The algorithm stops directly with a small $k_{max} = 15$, $j = 1$, $r = 2$, $Q_{min} = 1.0731\times10^{-24}$ and $Q_a = 1.5481\times10^{-21}$.

Here is the graph of $y_1 - y$:



Here is the graph of $\eta - y$ (η is the NDSolve solution), were we can see, that the numerical solution of NDSolve has very big deviations:



Example VI

We apply the algorithm on a second order ODE with boundary conditions:

$y'' = (y + y^2 - e^{-2t / sqrt(\mu)})/\mu$ with $\mu = 1/100$ and with $y(0) = 1$, $y(1) = e^{-1/sqrt(\mu)}$, the approximation interval is $I = [0, 1]$.

We called the module for the algorithm with:

```
WCollocationS2Alg[-100 (-e^-20t+y[t]+y[t]^2) + y''[t], {0,1}, {1,1/e^10},0.,1.,
15,1,2,2]
```

Mathematica NDSolve has problems:

```
NDSolve::ndsz: At _t_ == _0.9415179282009`_, step size is effectively zero;
singularity or stiff system suspected. ≫
General::stop: Further output of _NDSolve::ndsz_ will be suppressed during this
calculation. ≫ Divide::infy: Infinite expression _-(2.07326×10^-289/0.)_ encountered.
```

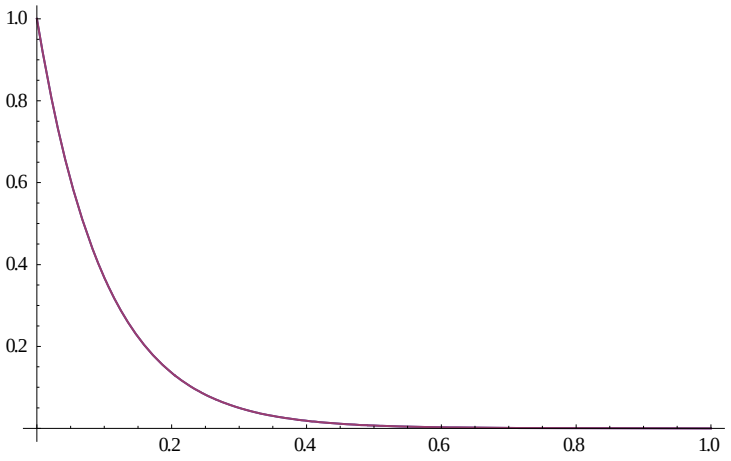
Mathematica automatically quits the kernel. The algorithm has no problems.

Now we see the iteration-protocol:

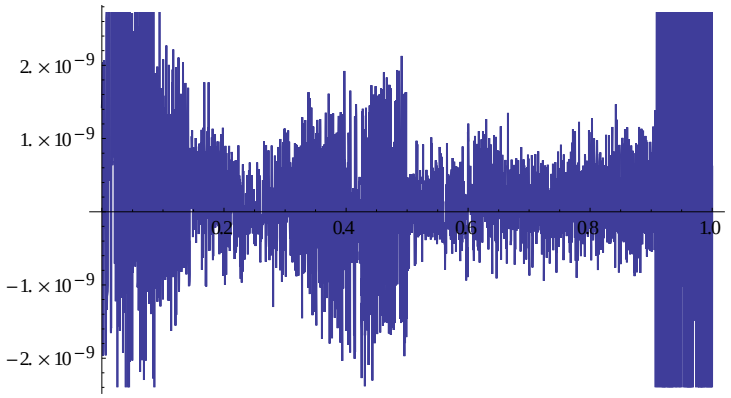
$k_{max}^{(0)}$	J	r	Q_{min}	Q_a
15	1	2	8.24021×10^{-14}	7.66528×10^{-11}

For critical examples we could start with a higher k_{max}, j and r.

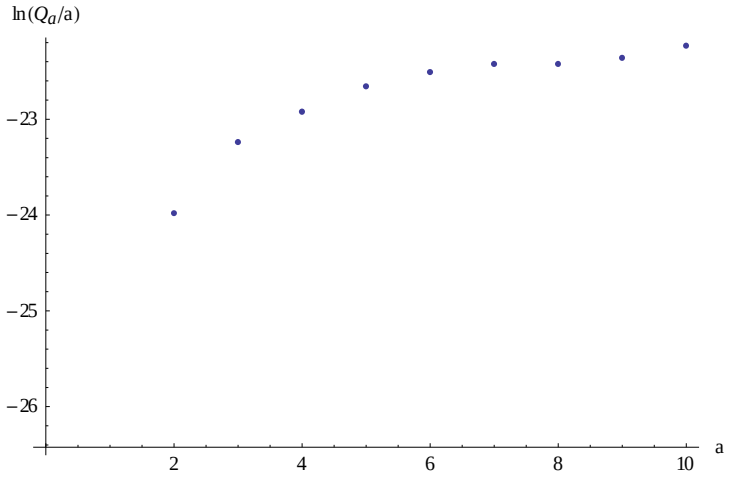
Here are the graphs of y_4 and y (we see no differences):



Here is the graph of $y_4 - y$:



At last we see graphically the relation between a und Q_a/a in this example:



References

[1] Abdella, K. (2012). "Numerical Solution of Two-Point Boundary Value Problems Using Sinc Interpolation", *Proceedings of the American Conference on Applied Mathematics (American-Math '12): Applied Mathematics in Electrical and Computer Engineering*

[2] Ascher, U. A. Mattheij, R. M. M. Russell, R. D. (1988). „Numerical Solution of Boundary Value Problems for ODEs", *Prentice Hall (Series in Computational Mathematics)*

[3] Ascher, U. Christiansen, J. Russell, R. (1981). "Collocation Software for Boundary Value ODEs", *ACM Trans. Math. Software*

[4] Bertoluzza S. (2006). "Adaptive Wavelet Collocation Method for the Solution of Burgers Equation," *Transport Theory and Statistical Physics*

[5] Carlson, T. S. Dockery, J. Lund, J. (1997). "A Sinc-Collocation Method for Initial Value Problems", *Mathematics and Computation, Vol. 66, No. 217*

[6] Donoho, D. L.; (1992). "Interpolating Wavelet Transforms," *Tech. Rept. 408. Department of Statistics, Stanford University, Stanford*

[7] Hairer, E. Wanner, G. (1993). Vol. 1 : "Nonstiff Problems", *Springer 2. Auflage*

[8] Hairer, E. Wanner, G. (1996). Vol. 2 : "Stiff and Differential-Algebraic Problems", *Springer 2. Auflage*

[9] Mei, S.-L. Lv, H.-L. Ma, Q. (2008). „Construction of Interval Wavelet Based on Restricted Variational Principle and Its Application for Solving Differential Equations", *Hindawi Publishing Corporation Mathematical Problems in Engineering*

[10] Nurmuhammada, A. Muhammada, M., Moria, M. Sugiharab, M. (2005). "Double Exponential Transformation in the Sinc-Collocation Method for a Boundary Value Problem with Fourth-Order Ordinary Differential Equation," *Journal of Computational and Applied Mathematics*

[11] Qian, L. (2002). "On the Regularized Whittaker-Koltel'nikov-Shannon Sampling Theorem", *Proceedings of the Amarican Mathematical Society, Vol. 131, No. 4*

[12] Robertson, H. H. (1975). "Some Properties of Algorithms for Stiff Differential Equations", *J. Inst. Math. Applics.*

[13] Russell, R. D. Christiansen, J. (1979). "A Collocation Solver for Mixed Order Systems of Boundary Value Problems", *Mathematics of Computation*

[14] Schuchmann, M. (2012). "Approximation and Collocation with Wavelets. Approximations and Numerical Solving of ODEs, PDEs and IEs," *Osnabrück: DAV*

[15] Schuchmann, M. (2008). "Parameteridentifikation dynamischer Systeme auf günstigen Pfaden", *DAV*

[16] Schuchmann, M.; Rasguljajew, M. (2013). Error Estimation of an Approximation in a Wavelet Collocation Method. *Journal of Applied Computer Science & Mathematics, No. 14 (7) / 2013, Suceava*

[17] Schuchmann, M.; Rasguljajew, M. (2013). Parameter Identification with a Wavelet Collocation Method in a Partial Differential Equation. *Journal of Approximation Theory and Applied Mathematics (JATAM) Vol. 1*

[18] Schuchmann, M.; Rasguljajew, M. (2013). An Approach for a Parameter Estimation with a Wavelet Collocation Method. *Journal of Approximation Theory and Applied Mathematics (JATAM) Vol. 1*

[19] Schuchmann, M.; Rasguljajew, M. (2013). Approximation of Non $L^2(R)$ Functions on a Compact Interval with a Wavelet Base. *Journal of Approximation Theory and Applied Mathematics (JATAM) Vol. 2*

[20] Shi, Z.; Kouri, D.J.; Wei, G.W.; Hoffman, D. K.; (1999). „Generalized Symmetric Interpolating Wavelets", *Computer Physics Communications*

[21] Strang, G.; (1989). "Wavelets and Dilation Equations: A Brief Introduction", *SIAM Review Vol. 31, No. 4*

[22] Unser, M. (1996). "Vanishing Moments and the Approximation Power of Wavelet Expansions", *Proceedings of the 1996 IEEE International Conference on Image Processing*