Implementing the IAPWS-95 standard in MATLAB®

Peter Junglas

Private University of Applied Sciences Vechta/Diepholz/Oldenburg
Schlesierstraße 13a, 49356 Diepholz, Germany
Email: peter@peter-junglas.de

WATER95 is a program library for the MATLAB environment that implements the IAPWS-95 standard for the thermodynamic properties of water. It is written mainly for educational applications and is published under an open source license. The routines provided by WATER95 are organised in three layers: Low level routines directly implement the formulae of IAPWS-95, basic user routines allow computation of the fundamental state equations and high level routines provide additional functions mainly for the needs of thermodynamics lecturers.

Special care has been taken to implement the most critical step in the IAPWS-95 formulation, namely the computation of the pressure and densities on the saturation curve. The necessary two step iteration process is described in some detail. In addition some effort is necessary to accelerate some of the routines and make the computation times feasible. Corresponding implementation details are presented together with some performance figures.

The results of WATER95 have been tested against published values and show perfect agreement, generally to all digits.

Introduction

The precise computation of the thermodynamical properties of water and steam is of paramount importance for many scientific and industrial applications. For this purpose the IAPWS has released two different formulations, which reproduce the measured properties of water within the known accuracy:

- IAPWS-95 [1,2], the current standard for general and scientific use,
- IAPWS-IF97 [3,4], designed for the special requirements of the steam power industry, which is easier to implement and generally faster than IAPWS-95.

IAPWS-95 is based on one formula for the specific free energy $f(\rho, T)$ over the complete relevant state space, all other thermodynamical quantities are derived from it. This approach automatically guarantees the internal consistency of the results, as long as the free energy behaves reasonably. But the explicit computation of the state variables requires function inversions, which have to be done iteratively and slow down the computations considerably.

The IAPWS-IF97 approach is completely different: The state space is divided into five regions defined by pressure and temperature boundaries, one of them identical to the saturation curve. In each region a formula either for the specific free energy $f(\rho, T)$ or for the specific Gibbs free energy $g(p, T)$ is given. To avoid the necessity of function inversions additional explicit formulae are stated for inverse equations ("backward equations") such as $T(p, h)$ and $T(p, s)$. The consistency of the formulation when crossing the region boundaries or using the backwards equations has been carefully checked.

Since the design of the IAPWS-IF97 formulation allows for fast, straightforward implementations, ready to use programs are available for many environments, e.g. in FORTRAN [5], in C++ [6], in JavaScript [7], for MS-Excel [8] and for several pocket calculators [9]. Some of these programs are open source. For IAPWS-95 two commercial software packages exist ([10], [11]), the last one is free for university use, but not open source. The freesteam project [6] contains some basic IAPWS-95 routines, but is still very rudimentary.

WATER95 [12], the program library presented here, is an open source implementation of IAPWS-95 for the MATLAB environment. It has been designed for educational purposes, but should be useful for general applications, which are not time-critical and want to employ IAPWS-95. Some simple examples like the creation of water and steam tables, the plotting of isobaric or adiabatic processes or the computation of the efficiency of the Clausius-Rankine cyclic process, have been described in [13].

Since a high-level mathematical problem solving environment like MATLAB provides a plethora of general purpose algorithms, one can concentrate on the specific numerical issues related to the implementation of IAPWS-95. Furthermore MATLAB is widely-used in educational as well as industrial contexts. As the source of WATER95 is freely available, one can study and possibly enhance the numerical algorithms applied.

After a short summary of the basic ideas of IAPWS-95 we will present in the following the general design of the library, give some implementation details with a special focus on the computation of the saturation pressure and show the results of correctness and performance tests.

The IAPWS-95 standard

In the following the basic steps of the IAPWS-95 standard are summarised for convenience. They are taken from [2] and are extensively described and discussed in [1].

The IAPWS-95 formulation is based on an explicit formula for the specific free energy *f*(*ρ*, *T*) of water as a function of the density $\rho = 1/v$ and the temperature *T*. The functional form of *f* is the result of mathematical considerations and physical insight, its many coefficients are the outcome of a very complicated iteration scheme, which is used to fit the formula to all experimental data available.

The state equations can be easily computed using derivatives of the free energy, e. g.

$$
\begin{aligned}
p(\rho,T) &= \rho^2 \frac{\partial f}{\partial \rho} \\
h(\rho,T) &= f - T\frac{\partial f}{\partial T} + \rho\frac{\partial f}{\partial \rho} \qquad (1) \\
s(\rho,T) &= -\frac{\partial f}{\partial T}
\end{aligned}
$$

If the values of *ρ* and *T* lie inside the two-phase region (wet steam) the formula for the free energy given in IAPWS-95 produces unphysical results, as can be seen in a p-v diagram (Fig. 1).

As a consequence of thermodynamic stability, the pressure as a function of the volume should decrease, but it oscillates instead. To cope with this problem one has to use the well-known Maxwell construction, which gives the values of the saturation pressure p_S and the specific volumes *v*' and *v*'' of pure water and pure vapor: p_S has to be chosen such that the areas of the graph of *p*(*v*) below and above the line $p = p_S$ are equal, i.e.

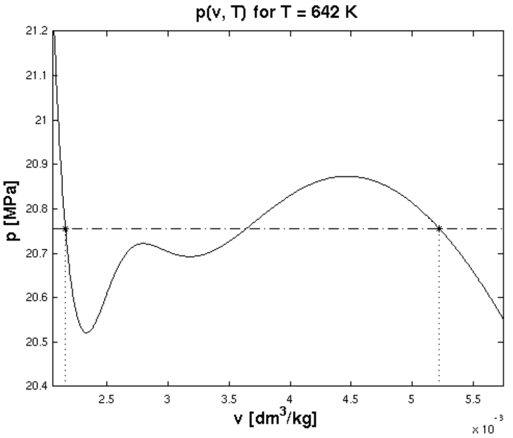

Figure 1: Example of the Maxwell construction

$$p_S(T)(v'' - v') = \int_{v'}^{v''} p(v,T)\,dv \qquad (2)$$

Formulated in terms of the densities this amounts to simultaneously solving the following equations for $p_S(T)$, *ρ*' and *ρ*'':

$$
\begin{aligned}
p_S(T)\left(\frac{1}{\rho''} - \frac{1}{\rho'}\right) &= f(\rho',T) - f(\rho'',T) \\
p(\rho',T) &= p_S(T) \qquad (3) \\
p(\rho'',T) &= p_S(T)
\end{aligned}
$$

By adding this construction to the equation for the free energy the IAPWS-95 formulation reproduces the wet steam properties as well. In the following the values that result from direct application of *f*(*ρ*, *T*) without employing the Maxwell construction, will be called *raw values* and be denoted with an index *raw*.

Design of the WATER95 package

The routines provided by WATER95 are organised in several layers: The low level routines implement the IAPWS-95 equations, they are defined as *private* and can not be accessed directly. All other routines can be used after the package directory has been added to the MATLAB path. The basic user routines define the state equations, whereas higher level functions are provided for the standard tasks that are needed in a thermodynamics course.

The basic private routine is `freeEnergyRaw`, which reproduces the complicated formula for *f*(*ρ*, *T*) without employing the Maxwell construction. Similar raw routines are given for pressure, enthalpy and entropy; they implement formula (1) and define arguments and return values in standard

SI units. They are based on auxiliary functions `phi0`, `phir` and their derivatives, which all take dimensionless arguments. A last group consists of auxiliary functions that provide good starting values for computing the phase boundaries.

The user level routines `pressure`, `enthalpy` and `entropy` allow to compute the physically correct state equations of water as functions of ρ and T. If the state lies in the wet steam region, they additionally return the dryness fraction

$$x = \frac{m_{\text{vapor}}}{m_{\text{vapor}} + m_{\text{water}}} \qquad (4)$$

To identify the two-phase region they all rely on the function

```
[pS,rhop,rhopp] = saturationPressure(T)
```

that uses the Maxwell scheme to compute the saturation pressure p_S and densities ρ', ρ'' of the liquid and vapor phase. Additional functions supply the basic inverse relations - the saturation curve $T(p_S)$, $\rho(p, T)$ and $T(p, \rho)$ - and some other often needed inversions like $T(p, s)$ or $x(T, s)$.

Four groups of high-level routines that show how the basic functions can be applied are provided mainly for the needs of thermodynamics lecturers:

- functions to compute the state variables along an isentropic, isobaric, isochoric or isothermal,
- computations related to the Clausius-Rankine cyclic process,
- a plot function that displays the results of the routines above in a *p-V* and a *T-S* diagram as well as some more specialised plotting routines,
- routines to create standard tables for water, wet and hot steam.

Some simple examples that show how to use WATER95 in applications and how to extend its functionality can be found in [13]. The package and a complete list of all functions can be downloaded from the WATER95 homepage [12]. It is provided under standard open source conditions.

Computation of the saturation pressure

The most critical step in the IAPWS-95 formulation is the computation of the pressure and densities on the saturation curve using formula (3). The basic iteration step can be immediately formulated in MATLAB as

```
pS = (freeEnergyRaw(rhop,T) -
freeEnergyRaw(rhopp,T))/(1/rhopp - 1/rhop);
   rhop = density(pS, T);
   rhopp = density(pS, T);
```

The routine `density` inverts `pressureRaw` by using MATLAB's zero finding function `fzero`, which is the basic tool for most of the difficult parts of all following computations. If `fzero` is provided with a start interval, which brackets a zero (i.e. the function has different signs on the two end points of the intervall), it uses the Dekker-Brent algorithm [14] to find the zero. Unfortunately such an interval is seldom known beforehand. Alternatively one can call `fzero` with a simple start value, in which case it tries to find a bracketing interval by gradually increasing a small start interval around the first guess. This can lead to big trouble, if the growing search interval leaves the domain of the function or - even worse - encircles three or more zeros and converges to a wrong one.

For values well below the critical point T_c, the raw pressure as a function of the inverse density (i.e. the specific volume) oscillates wildly. Therefore bad starting values for the iteration can lead to wrong solutions. For example, at $T = 373$ K, the results of using start values for the densities, which are only 10 % off from the correct results, are shown in Table 1.

Table 1: Iteration results for bad start values

Variable	Correct value	Start value	Result of iterations
p_S [bar]	1.0088	1.0088	1.1807
ρ' [kg/m^3]	958.5	871.3	778.3
ρ'' [kg/m^3]	0.5952	0.6546	0.6986

The reason for this behaviour can be seen in Fig. 2, which uses a highly nonlinear scaling to cope with the large pressure variations, which actually oscillate by more than 10^{12} MPa.

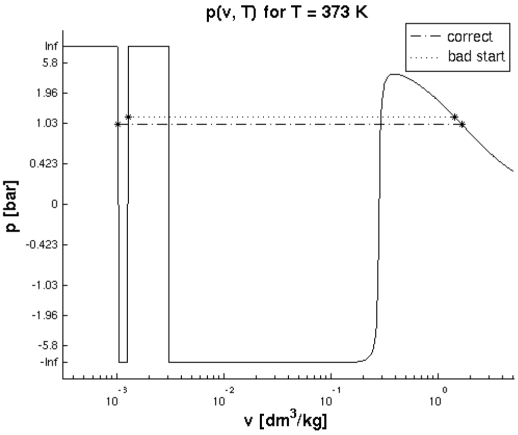

Figure 2: Maxwell construction at 373 K

The chosen start values lead to a solution, which indeed fulfils the Maxwell condition. But it doesn't include the narrow negative peak at low specific volume values. To get the correct result it is therefore essential to have good start values. Equations for the saturation pressure and densities are given in [15]. They are already very accurate, but can not be used here by themselves because they are not consistent with the IAPWS-95 formulation. Using them as start values, as proposed in [1], leads to correct results of the iteration for temperatures from the triple point up to values close to the critical temperature T_c.

Numerical experiments show that this scheme works up to 3.5 mK below T_c. Closer to T_c the simple iteration scheme fails: Starting with the good approximation $\rho' \approx 333.26$ kg/m^3 for T = T_c – 3.4 mK , the next iterations find the intermediate value at 321.8 kg/m^3 (see Fig. 3), after that the iteration diverges.

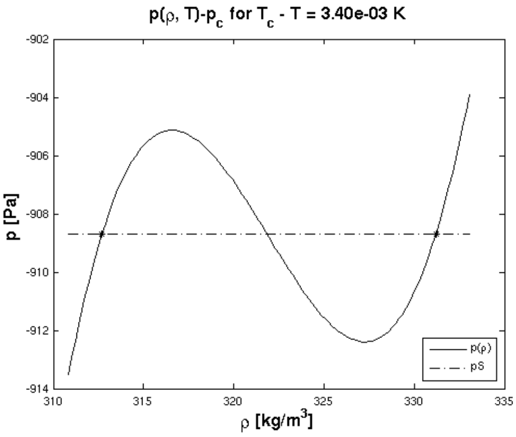

Figure 3: Maxwell construction at T_c – 3.4 mK

Obviously `fzero` closes in on the wrong (intermediate) root. To prevent that a lower bound for ρ' has to be given. Since close to T_c the function $p_{\mathrm{raw}}(\rho, T)$ behaves much more reasonably - like all examples of the Maxwell construction shown in textbooks it has only two local extrema -, one can use the MATLAB function `fminbnd` described in [14], to find the relative minimum that screens ρ' from the spurious solution. This provision (and a corresponding one for ρ'') lets the iteration converge on the correct values.

Approaching the critical point closer still (T_c – T < 6 μK), even this scheme breaks down: The minimum is so shallow that the approximate saturation pressure (though correct to 11 digits) lies below the minimum (see Fig. 4).

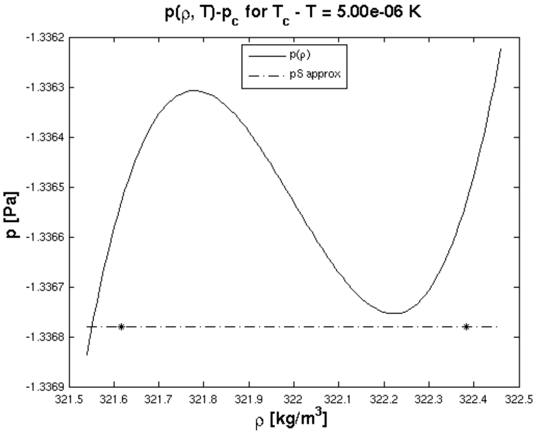

Figure 4: Maxwell construction at T_c – 5 μK

At this point, the theory of critical behaviour comes to the rescue: Near the critical point $p_S(T)$ and $\rho(T)$ should be dominated by power laws

$$p_c - p_S(T) \approx a(T_c - T) \qquad (5)$$

$$\rho_c - \rho(T) \approx \pm b(T_c - T)^c \qquad (6)$$

The values computed so far show excellent agreement with this theory (see Fig. 5) and can be used to extract the coefficients a, b, c by simple fits.

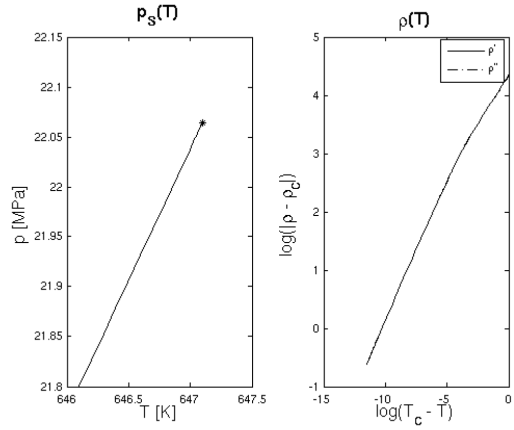

Figure 5: Critical behaviour of p_S and ρ'/ρ''

Results from equations 5 and 6 can now be put into one further iteration step or used directly, where the iteration doesn't work any more (at T_c – T < 50 nK). As Fig. 6 shows, we are now very close to the limits of the numerical precision. Any further refinements would need a lot of effort which is completely wasted in view of the accuracy of the measurements that form the basis of the IAPWS-95 formulation.

Further implementation details

All the numerical coefficients that are defined in the IAPWS-95 formulation are read in from an

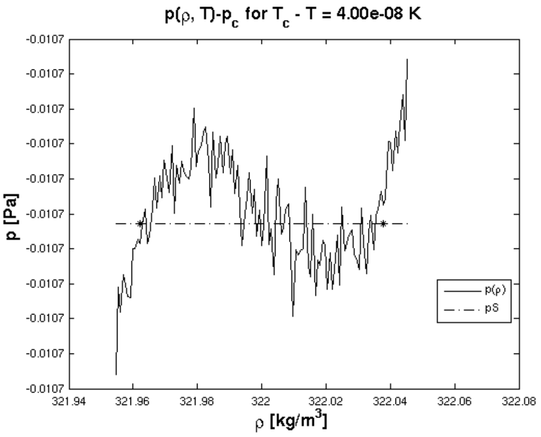

Figure 6: Maxwell construction at $T_c - 40$ nK

ASCII text file by `readIAPWS95data` and returned as a MATLAB cell array. The raw functions like `freeEnergyRaw`, `entropyRaw` and `pressureRaw` use a lazy initialisation scheme to read the coefficients when necessary. As proposed in [2], the values of n_1^0 and n_2^0 have been slightly adjusted to guarantee that at the triple point the standard values are exactly reproduced, i.e.

$$u'(T_t) = 0, \quad s'(T_t) = 0 \qquad (7)$$

All inverse functions are implemented with `fzero`. In most cases a simple start value works for all input parameters, but for `density` and `temperaturePS` better estimates have to be used that depend on the input parameters. They are found by some adhoc computations which are the outcome of numerical parameter studies. All inverse functions have an optional parameter to provide a reasonable start value, if one is known beforehand. This is especially useful for functions that describe state changes (isobaric, isochoric etc.): Utilising the result of the last step as starting value for the next one can considerably speed up the computation (cf. below).

The state variable routines `pressure`, `enthalpy` and `entropy` try to avoid the expensive computation of the saturation curve. For that purpose they use the approximation formula in `auxSaturationDensities` together with narrow error bounds. If the values ρ and T are outside the bounds the result is obtained from the corresponding raw function. Only if one is close to or inside the saturation region, a call to `saturationPressure(T)` is needed to give the exact values.

The basic routines `pressure`, `entropy` and `enthalpy` have simple cutoffs for temperatures below 0 °C. Though they work for lower temperatures as well, this trick makes some of the inversion routines much simpler. The cutoffs could probably be removed with some extra effort.

Testing the results

Besides the general problem of limited numerical accuracy, another potential source of errors are the function inversions implicit in the calculation of the saturation curve and explicit in the computation of the inverse state equations. In addition to some general consistency checks, the computed values of the thermal, caloric and entropic state equations have therefore been compared to published values from several sources. A special focus has been laid on values on the saturation curve near the critical point.

Checking with published tables

For a quick first check of an implementation of IAPWS-95 a few values with high accuracy - mostly nine digits - have been published in [2]. WATER95 reproduces most of the values exactly, five values differ by one in the last (ninth) digit. This difference is negligible for practical purposes due to the much larger errors of the actual measurements. It seems interesting though to find the reason of this discrepancy, since it may be an indication of potential numerical problems.

A more thorough test is the check against the values published in [1], which contains one table with 193 points on the saturation curve and another table with 2142 entries in a single phase region as well as 19 additional values on the saturation curve. Except for 38 points which are below $T = 0$ °C, all other values are reproduced by WATER95 to all published digits. If one removes the cutoff in the computation of the enthalpy - which, as mentioned above, will lead to convergence problems with some of the inverse functions - these 38 points are reproduced as well.

Testing with NIST values

Another source of values that are based on IAPWS-95 is the NIST Chemistry WebBook, a large online database of physical and chemical properties of many different substances, amongst them the thermodynamical properties of water [16]. A web interface provides access to isothermal, isobaric and isochoric values as well as to the saturation curve, displaying the values in form of plain tables with five significant digits or as an XY plot of any two variables.

A first test with a set of randomly chosen values scattered over the state space led to identical results. Differences have been found for the densities along the saturation curve very close to the critical point (cf. Table 2).

Table 2: Comparison of WATER95 and NIST WebBook values near T_c

T **[K]**	p_S **[MPa]**	***ρ*',W95** **[kg/m³]**	***ρ*',NIST** **[kg/m³]**	***ρ*'',W95** **[kg/m³]**	***ρ*'',NIST** **[kg/m³]**
647.01	22.041	356.07	355.85	287.78	287.56
647.05	22.052	349.50	349.32	294.37	294.20
647.08	22.060	340.39	340.27	303.46	303.35
647.09	22.062	333.96	333.90	309.90	309.84
647.095	22.064	327.18	327.16	316.80	316.79

The correct values of ρ' and ρ'' should satisfy

```
rhop = density(pS, T);
rhopp = density(pS, T);
```

The function `pressureRaw(rho, T)` is quite reliable (except for potential numerical problems), since it directly implements the IAPWS-95 functions without any inversions. Using it one can get a simple measure of the accuracy by

```
p1 = pressureRaw(rhop, T);
p2 = pressureRaw(rhopp, T);
eps1 = abs(p1 - p2)/p1
```

To check the WATER95 values against the NIST WebBook values fairly, one has to round them to the same precision. This leads to the error values shown in Table 3 (including the exact values for comparison).

Table 3: Relative pressure differences at ρ' and ρ''

T **[K]**	**W95** **exact**	**W95** **rounded**	**NIST**
647.01	5.070e-15	5.168e-09	1.218e-08
647.05	0.000e+00	7.186e-09	1.610e-08
647.08	3.546e-15	7.707e-10	5.315e-10
647.09	5.066e-16	9.200e-10	1.126e-09
647.095	2.364e-15	3.494e-11	2.385e-10

Though the (rounded!) WATER95 values seem to be a bit better, the differences are small. Another test the correct results have to fulfil is the validity of the Maxwell condition (cf. Fig. 7): The areas A_+, A_- of the graph of the raw pressure function $p_{\text{raw}}(v)$ as a function of the specific volume v above and below the saturation pressure p_S have to be equal.

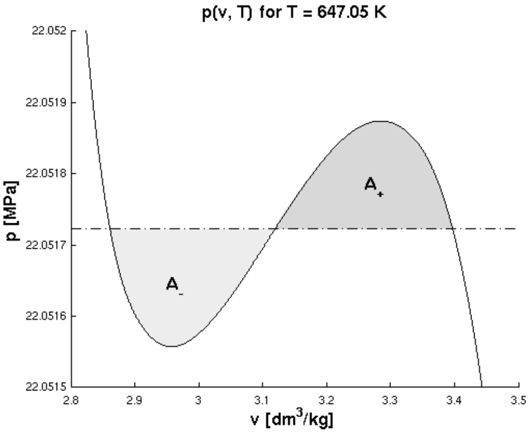

Figure 7: Definition of areas to compare

The areas can be computed easily with MATLAB's `quad` function and can be used to define another error indicator

$$\varepsilon_2 = \left|A_+ - A_-\right| / A_+ \qquad (8)$$

The results are given in Table 4 and show that the WATER95 values fulfil this requirement much better than the NIST WebBook values.

Table 4: Relative area differences at ρ' and ρ''

T **[K]**	**W95** **exact**	**W95** **rounded**	**NIST**
647.01	4.871e-09	5.493e-09	9.727e-05
647.05	1.001e-08	6.915e-08	4.268e-05
647.08	1.364e-07	1.802e-07	2.282e-05
647.09	9.649e-07	1.461e-06	3.715e-05
647.095	1.120e-05	1.010e-05	1.625e-05

Comparing with FLUIDCAL

Finally the values are compared to the outcomes of the program FLUIDCAL/WATER [11], which comes as a DLL library together with an interface to MS Excel®. A quick check of the FLUIDCAL results with the test values from [2] reveals three differences in the last digit, which coincide with the corresponding WATER95 values. This seems to indicate that the error may lie in the published values, which possibly are not rounded correctly.

Comparing the other two values that differed in WATER95 shows that the FLUIDCAL results coincide with the reference values in [2]. One of them lies in the liquid region, where numerical problems are expected. For the other one the apparent error of 10^{-9} is an artefact of the rounding, the actual relative difference between the two values is only 5×10^{-11}.

Finally FLUIDCAL has been used to compare the results very close to the critical point. The values almost coincide, the saturation pressures differ by 10^{-11}, the densities by 10^{-8} (cf. Table 5).

Table 5: Relative differences of WATER95 and FLUIDCAL results

T **[K]**	p_S	ρ'	ρ''
647.05	2.764e-11	4.960e-09	8.382e-09
647.08	2.654e-11	1.704e-08	1.467e-08
647.09	2.668e-11	3.092e-08	3.111e-08
647.095	2.667e-11	7.722e-08	7.822e-08
647.09599	2.745e-11	7.855e-06	9.571e-06

The error defined by the relative pressure differences is shown in table 6.

Table 6: Relative pressure difference at ρ' and ρ''

T **[K]**	**WATER95**	**FLUIDCAL**
647.05	0.000e+00	7.624e-12
647.08	3.546e-15	4.952e-12
647.09	5.066e-16	3.176e-12
647.095	2.364e-15	1.353e-12
647.09599	1.013e-15	1.496e-13

The values for WATER95 seem to be better, almost at the limit of what can be achieved with double precision arithmetic. This is confirmed by the respective plot of the Maxwell construction shown in Figure 8.

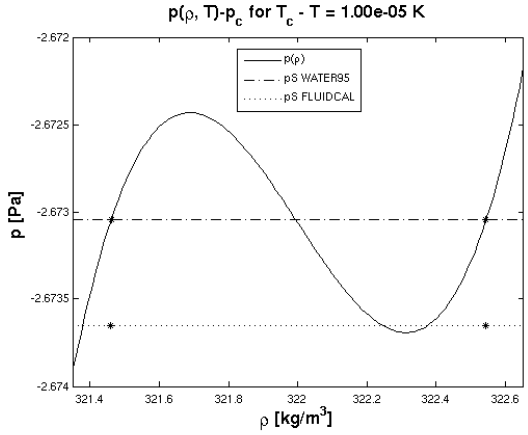

Figure 8: Comparison of a Maxwell construction for WATER95 and FLUIDCAL

However for a conclusive comparison, one has to plot the results of FLUIDCAL against its own computations of the raw pressure to check its internal consistency. Unfortunately these values are not available from the software.

Performance of WATER95

A number of simple benchmark tests have been run to measure computation times for a few standard tasks. The following results have been obtained on a standard PC with Pentium-D 2x3.40 GHz cpus running openSUSE 10.3 (kernel version 2.6.22.17-0.1) and MATLAB R2007b. Each benchmark has been run four times consecutively, the execution times have been measured with MATLAB's `tic/toc` commands. The results summarized in table 7 show the mean values of the last three runs.

Table 7: Results of WATER95 benchmarks

benchmark	**time [s]**
water table	6.8
hot steam table	5.3
wet steam table	23.6
isobaricSimple	66.3
isobaric	10.1
isochoric	26.5
isothermal	2.5
isentropicTPP	15.3
clausiusRankine	86.2

The first set of tests covered the computation of simple tables for water, wet and hot steam as they can be found in thermodynamics textbooks. The water table contains 134 state points with pressure *p* ranging from 0.2 bar to 200 bar and temperatures *T* from 0 °C up to 360 °C (for high pressure). The hot steam table has 268 rows containing values from 0.1 bar/50 °C up to 500 bar/800 °C. The wet steam table has 141 temperature values spanning the complete saturation curve. These tests can be reproduced by the commands

```
[p, T, v, h, s] = getWaterTable();
[p, T, v, h, s] = getHotSteamTable();
[p, T, v, h, s] = getWetSteamTable();
```

The second set of benchmarks consisted of the computation of an isobaric, isochoric, isothermal and isentropic state change, each crossing the saturation line, with the following parameter values:

```
[pi,vi,Ti,si,xi] =
    isobaric(40e5,1e-3,0.086442);
[pi,vi,Ti,si,xi] =
    isochoric(3.5e3,0.2,13.5e5);
[pi,vi,Ti,si,xi] =
    isothermal(200e5,1.361e-3,0.1,100,30);
[pi,vi,Ti,si,xi] =
    isentropicTPP(550,10e5,1e5,100,1);
```

To show the importance of performance tuning, the isobaric has been computed additionally with a simple version that doesn't reuse intermediate values as start values.

Finally a complete Clausius-Rankine cyclic process with 400 points has been computed with

```
[pi,vi,Ti,si,xi, N] =
      clausiusRankine(773,60e5,0.1e5,1);
```

The results show that though WATER95 surely is fast enough as an educational tool, there is still a lot of potential for tuning. Using the MATLAB profiling utility one can easily analyse the timing behaviour of the last benchmark. This reveals that by far the most time is spent inversing `pressureRaw` to get the density in the computation of the saturation pressure. If performance should be an issue, an obvious starting point would therefore be to speed up this inversion process, e.g. by introducing faster, though less accurate auxiliary routines in earlier phases of the iteration.

Conclusions

The consistency tests show that the results of WATER95 on the saturation curve are very reliable and at least as accurate as the values computed with other available programs. In all other regions (except for $T < T_t$) WATER95 reproduces the published results. Its routines cover most of the topics that are discussed in standard thermodynamics lectures and are fast enough for teaching purposes.

For a fully-fledged implementation of IAPWS-95 and to comply with the requirements of non-educational applications, WATER95 should be enhanced in several ways:

- addition of routines for the second-order derivatives like heat capacity and speed of sound as well as a full set of inverse functions,
- extension of the domain to temperatures below the triple point,
- general improvement of the performance.

The first two points could certainly be implemented easily, but to speed up the inversion functions is a more difficult task. A possible starting point could be the introduction of sophisticated auxiliary functions. A more radical approach would be the substitution of the `fzero` routine by a different algorithm that uses the properties of the given functions. On the other hand such amendments would probably destroy the simplicity and educational character of the code.

If performance is really the issue - and the use of IAPWS-IF97 not the immediate solution -, one could consider porting the whole library to another programming language like Fortran or C++, either as a stand-alone program or as part of the freesteam open-source project [6].

Note added in proof:

The author has been informed that a minor rounding problem in FLUIDCAL and related software packages has been found recently. A new version of [2] is in preparation, which contains the corrected values. They agree with WATER95 results to all published digits.[1]

Literature

[1] *W. Wagner and A. Pruß:* The IAPWS Formulation 1995 for the Thermodynamic Properties of Ordinary Water Substance for General and Scientific Use. J. Phys. Chem. Ref. Data, 31, 387-535 (2002).

[2] *International Association for the Properties of Water and Steam:* Release on the IAPWS Formulation 1995 for the Thermodynamic Properties of Ordinary Water Substance for General and Scientific Use. IAPWS Release, IAPWS Secretariat (1996).

[3] *W. Wagner, J. R. Cooper et. al.:* The IAPWS Industrial Formulation 1997 for the Thermodynamic Properties of Water and Steam. J. Eng. Gas Turbines Power, 122, 150-182 (2000).

[4] *International Association for the Properties of Water and Steam:* Revised Release on the IAPWS Industrial Formulation 1997 for the Thermodynamic Properties of Water and Steam. IAPWS Release, IAPWS Secretariat (2007).

[5] *American Society of Mechanical Engineers:* Steam Properties for Industrial Use Based on IAPWS-IF97 Professional Version (2003). http://crtd.asme.org/wsts/steam.html

[6] *freesteam project:* http://freesteam.sourceforge.net/ (2007).

[7] *L. Kos:* IAPWS-IF97 implementation in JavaScript. Laboratory for Engineering Design, University of Ljubljana. (2005). http://www.lecad.uni-lj.si/~leon/research/iapws-if97/IAPWS97.html

[8] *H.-J. Kretzschmar, I. Stöcker et al.:* LibIF97 - Software for the Industrial Formulation IAPWS-IF97 for Water und Steam (2007).

[1] The author would like to thank the editor for pointing this out and providing the prospective new version of [2].

http://thermodynamik.hs-zigr.de/cmsfg/Stoffwertbibliothek/index.php

[9] *H.-J. Kretzschmar, I. Stöcker et al.:* Steam Tables on Pocket Calculators - Property Library for the Industrial Formulation IAPWS-IF97 of Water and Steam (2007). http://www.steamtables-pocket-calculators.com/

[10] *National Institute of Standards and Technology:* NIST/ASME Steam Properties (2007). http://www.nist.gov/data/nist10.htm

[11] *W. Wagner:* Software for the IAPWS-95 Formulation (2007). http://www.ruhr-unibochum.de/thermo/Software/Seiten/IAPWS-95-eng.htm

[12] *P. Junglas:* WATER95 homepage (2007). http://www.peter-junglas.de/fh/water95/index.html

[13] *P. Junglas:* WATER95 – A MATLAB® Implementation of the IAPWS-95 Standard for Use in Thermodynamics Lectures. To appear in Int. J. Engng. Ed.

[14] *C. B. Moler:* Numerical Computing with Matlab. Cambridge University Press, Cambridge (2007).

[15] *W. Wagner and A. Pruss:* International Equations for the Saturation Properties of Ordinary Water Substance. Revised According to the International Temperature Scale of 1990. J. Phys. Chem. Ref. Data, 22, 783-787 (1993).

[16] *National Institute of Standards and Technology:* Thermophysical Properties of Fluid Systems (2005). http://webbook.nist.gov/chemistry/fluid/