Recommended method for determining the correlated color temperature and distance from the Planckian locus of a light source

Kevin Smet[a]
Michael Royer[b,c]
Doug Baxter[d]
Eric Bretschneider[e]
Tony Esposito[f]
Kevin Houser[c]
Wendy Luedtke[g]
Kwong Man[h]
Yoshi Ohno[i]

[a] ESAT-WaveCore/Light&Lighting Laboratory, Ku Leuven, Ghent, Belgium
[b] Pacific Northwest National Laboratory, Portland, OR, USA
[c] School of Civil and Construction Engineering, Oregon State University, Corvallis, OR, USA
[d] Pacific Northwest National Laboratory, Richland, WA, USA
[e] EB Designs & Technology, USA
[f] Lighting Research Solutions, USA
[g] ETC, USA
[h] Signify, Vancover, BC, Canada
[I] National Institute of Standards and Technology, Gaithersburg, MD, USA

Corresponding Author: Michael P. Royer, Pacific Northwest National Laboratory, 620 SW, 5th Ave, Suite 810, Portland, OR 97204, USA. E-mail: michael.royer@pnnl.gov

This is an archival copy of an article first published online September 13, 2023, in *Leukos*. Please cite as

Smet K, Royer M, Baxter D, Bretschneider E, Esposito T, Houser K, Luedtke W, Man K, Ohno Y. Recommended Method for Determining the Correlated Color Temperature and Distance from the Planckian Locus of a Light Source. *Leukos.* Online first. DOI: 10.1080/15502724.2023.2248397

Abstract

Correlated color temperature (CCT) is one of the primary metrics used to characterize the visual appearance of light and is most informative when coupled with distance from the Planckian locus (Duv). Given a set of chromaticity coordinates—which are calculated from a light source's spectral power distribution—it is possible to compute CCT and Duv with varying levels of accuracy. Over the last six decades at least a dozen methods have been proposed to compute CCT while balancing accuracy with calculation complexity. CCT values become inconsistent at some level of precision when calculated using different methods, which in turn can lead to discrepancies in dependent or subsequent calculations used by lighting professionals and may be problematic in software. Although methods are now documented that can provide extremely high accuracy, no consensus body has recommended a preferred method. This analysis examines both the calculation speed and the CCT and Duv accuracy of previously proposed and new methods. With consideration of the calculation accuracy, computational burden, calculation complexity, and considerations of practical implementation, we recommend a recent modification of the Robertson method.

Keywords: CCT, Duv, Chromaticity, Calculation time, calculation accuracy

1 Introduction

Correlated color temperature (CCT) has long been one of the primary metrics by which lighting professionals evaluate light sources, but there is not an official calculation method recommended by consensus bodies such as the Illuminating Engineering Society (IES), National Electrical Manufacturers Association (NEMA), or International Commission on Illumination (CIE). IES defines CCT as "the absolute temperature of a blackbody whose chromaticity most nearly resembles that of the light source" (IES 2020a). CIE provides a more detailed definition, which reads in part: "Temperature of a Planckian radiator having the chromaticity nearest the chromaticity associated with the given spectral distribution on a modified 1976 UCS [Uniform Chromaticity Scale] diagram where u', $(2/3)v'$ are the coordinates of the Planckian locus and the test stimulus" CIE 2011). This CIE definition was updated in the 2004 version of CIE 15 (CIE 2018) to be purely a numerical definition—tied to a specific chromaticity diagram—without any implication that CCT represents the closest *visual* match to a Planckian radiator, as was previously part of the definition. Nonetheless, neither IES nor CIE explicitly recommends a calculation method to be used to determine CCT, despite reporting precise values for some standard illuminants (e.g., ISO 2022). CIE provides the following note with its definition of CCT, but this describes a concept, rather than a specific and actionable standard calculation method:

"Note 2 to entry: Correlated colour temperature can be calculated by a simple minimum search computer program that searches for that Planckian temperature that provides the smallest chromaticity difference between the test chromaticity and the Planckian locus, or for example by a method recommended by Robertson, A.R. "Computation of correlated color temperature and distribution temperature", J. Opt. Soc. Am. 58, 1528-1535, 1968. (Note that the values in some of the tables in this reference are not up to date.)"

Many methods for calculating CCT have been proposed. Early methods date to the 1930s (Davis 1931; Judd 1936), as the increasing availability of light sources with off-Planckian chromaticity required a method to characterize the relative appearance of the light. Despite being different from the methods that are familiar today, these early works helped formalize the concept of iso-temperature lines and a closest match (Harding 1950; Kelly 1963; MacAdam 1977; Wyszecki and Stiles 1982).

In 1968, Robertson defined a method for estimating CCT based on a lookup table (LUT) and interpolation formula in the CIE 1960 (u, v) UCS diagram (Robertson 1968). In 2014, Ohno proposed updates to the LUT and interpolation method to improve the accuracy of this approach, with two distinct options for the LUT depending on the desired accuracy (1% increments totaling approximately 300 rows or 0.25% increments totaling approximately 1,200 rows covering a CCT range of 1000 K to 20,000 K, with the exact number of rows depending on the treatment of the start and end conditions) (Ohno 2014). In the intermediate period, several researchers proposed methods based on a formula fit to the Planckian locus or chromaticity diagram (Schanda et al. 1978; Krystek 1985; Xingzhong 1987; McCamy 1992; Hernández-Andrés et al. 1999; Gardner 2000; Guo and Houser 2004), with varying degrees of success in improving the accuracy of calculations within a range of interest (Zheleznikova 2020; Prytkov and Kolyadin 2021). These prescriptive approaches require parameter refitting (i.e., recreation of the formulas) when using different color matching functions or a different UCS, while LUTs are more readily updated.

As computing resources have improved, iterative methods that provide an estimation of CCT based on an accuracy stopping criterion have been proposed. Li et al. (2016) proposed augmenting the Robertson

method with a Newton-Raphson (NR) algorithm (Ypma 1995). The latter is a root finding algorithm that finds progressively better roots of a function starting from an initial guess and the function's derivative. The additional algorithm can, in theory, provide accuracy up to any specified tolerance. Likewise, Zhang documented the performance of a *golden section* search method, which utilizes consecutively more refined LUTs—in this case reducing the size of the LUT using the golden ratio—to achieve a predetermined accuracy tolerance (Zhang 2019). More recently, Baxter, Royer, and Smet (2023) described how a Fibonacci search (Avriel and Wilde 1966; Overholt 1973) may improve the speed of searching a LUT, in conjunction with an interpolation method like those described by Robertson or Ohno, or as a standalone method. The presence of many alternatives means that choosing a preferred method for determining CCT cannot only be dependent on accuracy but should also consider computation speed. Speed matters, for example, in simulation analyses requiring millions or billions of calculations, or in control software when values must be determined rapidly in real time. Notably, nearly any level of accuracy can be achieved by altering the specific details of different methods but increasing the accuracy of a given method can increase calculation time and complexity.

For engineering and specification purposes, almost any of the proposed methods including and after Robertson's proposal likely provides sufficient accuracy for determining CCT. The lack of a single recommendation, however, is undesirable. For example, different versions of the method proposed by Ohno have been used in calculators supplied with ANSI/IES TM-30 (0.25% increment LUT) (IES 2020b) and CIE 224 (1% increment LUT) (CIE 2017)—neither written standard specifies the CCT calculation method used, nor is it readily apparent in the calculation software. This results in slightly different calculated CCT values, which in turn results in slightly different values for R_f—though the differences are negligible in normal use. The differences in both CCT and R_f are immaterial to practice, but consequential when debugging or validating software.

Duv (symbol: D_{uv}) is an important companion metric, such that CCT and Duv together provide an exact specification of chromaticity. This also compensates the incompleteness of CCT in conveying relative color appearance information (Durmus 2022). For some methods, such as those of Ohno, Duv is computed as part of the CCT calculation. In other cases, as when applying a Newton-Raphson algorithm, precise calculation of Duv is only possible after the final CCT is determined. Ohno also proposed a formula for approximating Duv based on chromaticity alone (Ohno 2014). Given that Duv is more straightforward to calculate, there is little existing literature investigating the accuracy of Duv calculations.

1.1 Scope

This study investigated the accuracy and speed of CCT and Duv calculations by implementing a selection of existing methods and new combinations. Each method's accuracy was determined by comparing estimated values for a reference set of chromaticity coordinates with known CCT and Duv values. Speed was assessed using independent implementations of the methods coded in Python and C. Together with subjective evaluations of calculation complexity and feasibility of implementation, results of the accuracy and speed assessments were used to reach a consensus recommendation.

The analysis was instigated by a working group of the Color Committee of the Illuminating Engineering Society to inform future decisions regarding standardization of a method for calculating CCT and Duv. A consensus process established a priori accuracy limits of 0.1 K maximum error for CCT and 0.0001 maximum error for Duv, which are an order of magnitude smaller than typical reporting practices and

should be sufficient for applied lighting purposes. While much higher accuracy is possible, the greatest benefit of a standardized method is the consistency and repeatability of the calculations, rather than accuracy beyond practical or scientific relevance. The committee also deemed it important to identify a method that would not be restricted to a specific set of color matching functions or chromaticity coordinate system.

2 Method

2.1 Determination of Reference Test Coordinates

Determining the accuracy of CCT and Duv calculations requires a set of reference values, which have traditionally been determined by converting a regular grid of CCT and Duv values to (*u*, *v*) chromaticity coordinates that fall along iso-temperature lines (Ohno 2014; Li et al. 2016; Zhang 2019), or at the bounds of the quadrangles specified in ANSI C78.377 (Prytkov and Kolyadin 2021). The reference coordinates are used as inputs to the CCT calculation method, such that the estimated CCT and Duv values can be compared to the index CCT and Duv values. Several methods have been proposed for computing iso-temperature lines and their associated chromaticity coordinates. Ohno (2014) proposed an "offset" approximation method (i.e., a numerical derivative method using the slope of the Planckian locus), and more recently, Prytkov and Kolyadin (2021) proposed a method based on the derivatives of the chromaticity coordinates with respect to the CCT. However, Prytkov and Kolyadin used numerical central derivatives based on finite differences in their calculations, and it can be shown that this method is equivalent to Ohno's offset method, except that the latter is based on forward or backward numerical derivatives. Decades ago, Robertson (1968) discussed a more accurate method, first proposed by Mori et al. (1964), that is based on analytical derivatives of a Planckian radiator, which therefore does not depend on the choice of the size of the "offset" or the finite differences. The difference between numerical and analytical derivative-based methods is substantially smaller than the CCT accuracy tolerance chosen for this work. Nonetheless, the exact analytical derivative method described by Mori et al. and Robertson was used to determine the reference coordinates. Equations are available in those publications.

The extents and resolution of the reference grids used in past work have varied. The extents in both CCT and Duv are critical in determining the maximum error for non-iterative methods, with resolution expected to only influence mean error. For this analysis, we considered the range of 1500 K to 40,000 K, with a Duv range of -0.05 to 0.05, excluding points that fall outside the spectrum locus. This range fully covers the characteristics of nominally "white" light sources typically employed in the built environment.

While in the past many studies adopted a reference set with a regular interval (usually in terms of reciprocal megakelvin), a preliminary analysis found that the interaction of the interval of a given LUT and the sampling of CCT and Duv in the reference set can bias the resulting accuracy determination. For example, a LUT with a resolution of 100 K would produce zero error if the evaluated reference CCTs were always in increments of 100 K, meaning that no computation would be required beyond the table lookup. To avoid such bias, a reference set was constructed by first creating an intermediate grid of CCT and Duv targets with values within the range of CCT and Duv specified above. Forty CCT targets were chosen: 1500, 1550, 1600, 1650, 1700, 1750, 1800, 1850, 1900, 1950, 2000, 2100, 2200, 2300, 2400, 2500, 2600, 2700, 2800, 3000, 3250, 3500, 3750, 4000, 4250, 4500, 4750, 5000, 5500, 6000, 6500, 7000, 8000, 9000, 10,000, 12,500, 15,000, 20,000, 30,000, and 40,000 K. Five Duv targets were chosen: -0.050, -0.025, 0.000, 0.025, and 0.050. Next, for each target grid coordinate (CCT_i, Duv_i), ten random (*CCT, Duv*) value pairs were generated within the following intervals: $[(X_{i-1} - X_i)/2, (X_{i+1} - X_i)/2]$, with *X* representing

either *CCT* or *Duv*. With the appropriate selection of CCT targets, this results in a set of (CCT, Duv) pairs that are also approximately uniformly distributed in the CIE 1960 UCS. The final 1,270 tested coordinates are shown in Figure 1, with the numerical data provided in the supplemental data file.

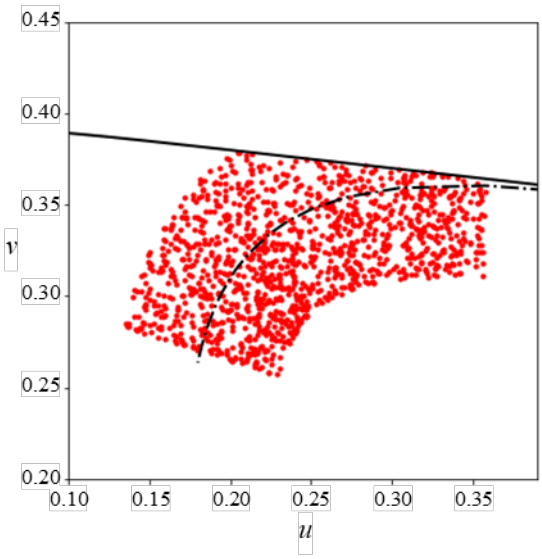

Figure 1. CIE 1960 UCS coordinates of the reference set (1,270 points).

2.2 CCT Calculation Methods

The final analysis reported here includes 40 methods, as summarized in Table 1. They are variations of three base approaches: Robertson (R) (Robertson 1968), Ohno (O) (Ohno 2014), and Fibonacci (F) (Baxter et al. 2023), with different calculation options, all of which meet the accuracy requirements specified in Section 1.1. New LUTs were created and used for these methods, replacing the original LUTs described in their references (e.g., Robertson [1968] used obsolete values in the original tables). The methods have LUTs with different increments: fixed kelvin increments (0.2 K, 1 K, 5 K, 10 K, 25 K, 50 K) or percentage change increments (0.25%, 0.5%, 0.75%, 1%). Not all increments are reported for all base approaches, but several hundred combinations (including additional LUT increments: 100 K, 250 K, 500 K, 1000 K, 5%, 15%) were examined in preliminary analyses, as described in Section 2.2.1. Note that for the Ohno-based methods, the triangular-to-parabolic transition point was not updated for each LUT, while the CCT-based correction factor was. Instead, for the former the previously published value of 0.002 was used, though we note that fine tuning this transition point can marginally reduce the maximum error. With optimized parameters, the Ohno method can meet the accuracy criteria with a LUT of 0.2%, but this variation is not further considered in this report because it does not have speed or data-size advantages.

Twenty-eight of the reported methods include a Newton-Raphson algorithm to improve upon the initial estimate, as described by Li et al. (2016) When a Newton-Raphson algorithm was applied, it included a stopping criterion of 0.1 K. However, the final accuracy was better by roughly 4 orders of magnitude (or better) due to the marked improvement possible with just one iteration.

Table 1. Description of reported methods, including the maximum CCT and Duv errors found in this analysis. All LUTs were generated from 1,000 K to 40,000 K. The abbreviation for each method is: Primary Algorithm (R, O, or F) [LUT Increment, Secondary Algorithm (NO for None or NR for Newton-Raphson)].

	LUT Increment	LUT Rows	Primary Algorithm	Secondary Algorithm	Abbreviation	Maximum CCT Error	Maximum Duv error
1	50 K	781	Robertson	None	R[50K,NO]	7.50E-02	7.70E-06
2	50 K	781	Robertson	Newton-Raphson	R[50K,NR]	8.90E-06	2.90E-08
3	25 K	1,561	Robertson	None	R[25K,NO]	1.80E-02	1.90E-06
4	25 K	1,561	Robertson	Newton-Raphson	R[25K,NR]	5.30E-07	2.20E-09
5	10 K	3,901	Robertson	None	R[10K,NO]	2.90E-03	3.10E-07
6	10 K	3,901	Robertson	Newton-Raphson	R[10K,NR]	1.40E-08	4.30E-11
7	5 K	7,801	Robertson	None	R[5K,NO]	7.20E-04	7.70E-08
8	5 K	7,801	Robertson	Newton-Raphson	R[5K,NR]	1.30E-08	1.30E-12
9	1 K	39,001	Robertson	None	R[1K,NO]	2.80E-05	3.10E-09
10	1 K	39,001	Robertson	Newton-Raphson	R[1K,NR]	1.40E-08	4.50E-15
11	1 %	372	Robertson	None	R[1%,NO]	9.70E-02	1.20E-06
12	1 %	372	Robertson	Newton-Raphson	R[1%,NR]	8.50E-07	2.40E-09
13	0.75 %	495	Robertson	None	R[0.75%,NO]	5.00E-02	6.60E-07
14	0.75 %	495	Robertson	Newton-Raphson	R[0.75%,NR]	2.60E-07	1.70E-10
15	0.50 %	741	Robertson	None	R[0.5%,NO]	2.50E-02	3.00E-07
16	0.50 %	741	Robertson	Newton-Raphson	R[0.5%,NR]	6.00E-08	1.60E-10
17	0.25 %	1,479	Robertson	None	R[0.25%,NO]	6.00E-03	7.40E-08
18	0.25 %	1,479	Robertson	Newton-Raphson	R[0.25%,NR]	1.80E-08	1.10E-11
19	50 K	781	Ohno	Newton-Raphson	O[50K,NR]	9.60E-09	1.20E-14
20	25 K	1,561	Ohno	Newton-Raphson	O[25K,NR]	1.30E-08	5.60E-16
21	10 K	3,901	Ohno	Newton-Raphson	O[10K,NR]	6.70E-09	3.60E-15
22	5 K	7,801	Ohno	None	O[5K,NO]	3.50E-02	4.50E-07
23	5 K	7,801	Ohno	Newton-Raphson	O[5K,NR]	6.20E-07	1.90E-08
24	1 K	39,001	Ohno	None	O[1K,NO]	7.70E-03	1.20E-08
25	1 K	39,001	Ohno	Newton-Raphson	O[1K,NR]	3.80E-08	3.00E-11
26	1 %	372	Ohno	Newton-Raphson	O[1%,NR]	1.70E-08	1.50E-13
27	0.75 %	495	Ohno	Newton-Raphson	O[0.75%,NR]	1.30E-08	1.10E-14
28	0.50 %	741	Ohno	Newton-Raphson	O[0.5%,NR]	7.80E-09	5.60E-16
29	0.25 %	1,479	Ohno	Newton-Raphson	O[0.25%,NR]	1.70E-08	6.30E-16
30	50 K	781	Fibonacci	Newton-Raphson	F[50K,NR]	9.70E-09	8.50E-12
31	25 K	1,561	Fibonacci	Newton-Raphson	F[25K,NR]	5.90E-09	2.80E-14
32	10 K	3,901	Fibonacci	Newton-Raphson	F[10K,NR]	2.70E-06	3.20E-09
33	5 K	7,801	Fibonacci	Newton-Raphson	F[5K,NR]	1.70E-07	3.30E-10
34	1 K	39,001	Fibonacci	Newton-Raphson	F[1K,NR]	1.80E-08	1.10E-12
35	0.2 K	195,001	Fibonacci	None	F[0.2K,NO]	1.00E-01	4.90E-07
36	0.2 K	195,001	Fibonacci	Newton-Raphson	F[0.2K,NR]	1.60E-05	4.90E-07
37	1 %	372	Fibonacci	Newton-Raphson	F[1%,NR]	1.70E-08	1.20E-15
38	0.75 %	495	Fibonacci	Newton-Raphson	F[0.75%,NR]	2.20E-08	9.40E-16
39	0.50 %	741	Fibonacci	Newton-Raphson	F[0.5%,NR]	5.60E-09	5.80E-16
40	0.25 %	1,479	Fibonacci	Newton-Raphson	F[0.25%,NR]	6.00E-09	5.50E-16

In accordance with CIE guidelines (CIE 2018), all LUTs were determined with Planckian radiation calculated from 360 nm to 830 nm in 1 nm increments, with constants $c_1 = 3.74177185 \times 10^{-16}$ W·m², $c_2 = 1.4388 \times 10^{-2}$ m·K, and index of refraction $n = 1$—though c_1 does not affect color calculations. These conditions and the constant values were used consistently in this analysis, as changing the wavelength range or intervals or the parameters c_2 or n would change the reference (u, v) coordinates. The LUTs extended from 1000 K to 40,000 K—1000 K being below the lowest test coordinate at nominally 1500 K.

2.2.1 Preliminary Analyses

The final group of 40 methods was selected through an iterative, exploratory process, where refinements were made to the specific implementations of the methods. In total, several hundred different methods were tested (not including different reference coordinate tests, different programming languages, etc.). The full gamut of methods comprised different combinations of base approach, LUT increment, secondary improvement algorithm (none, Newton-Raphson, or cascading

LUT—a method whereby new LUTs are created between the two closest estimates of another method—and repeated several times). The final group of 40 methods was analyzed with and without batch processing (i.e. converting multiple chromaticity coordinates in a single call to the conversion function) for the Python-based calculations (batches of 25 or no batches), but only results without batch processing are reported here because it did not change the recommended combination of base method and LUT. Nonetheless, the use of batch processing in Python resulted in substantial decreases in computation time for all methods due to a reduction of the size and memory requirements of some of the arrays created by the highly vectorized code. Formulaic methods, such as that of McCamy (1992), were excluded from the analysis based on previously demonstrated inferiority in accuracy (Prytkov and Kolyadin 2021), as well as the difficulty to adapt the method to other color matching functions or coordinate systems. Developing a method that is not restricted to a specific set of color matching functions or coordinate system was deemed an important consideration by the committee that initiated this work.

Methods that were initially considered but not reported here were not explored further due to poor performance in speed or accuracy. For example, the method of Zhang (2019) utilizing cascading LUTs based on a golden section search was found to be considerably slower than other alternatives at the desired tolerance level, regardless of the resolution of the original LUT. Cascading LUTs in general resulted in longer calculation times. Likewise, a recent alteration of the Ohno method by Li et al. (2022) that uses a spline fit instead of a parabolic fit was tested but was found to offer no significant advantages—it was slightly slower than the original Ohno method, and while the median error was improved, the maximum error was not substantially different.

Without a secondary improvement algorithm, base methods (Robertson, Ohno, or Fibonacci) with a 1000 K, 500 K, 250 K, 100 K, 15%, and 5% increment LUT could not reach the required 0.1 K accuracy for all 1,270 reference coordinates, so they were also omitted from Table 1. The original 30-row, variable-increment LUT of Robertson also did not meet the accuracy thresholds (max CCT error = 151.7 K), nor did either of the methods described by Ohno (1% LUT max CCT error = 2.6 K; 0.25% LUT max CCT error = 0.4 K). Only methods that met the 0.1 K CCT and 0.0001 Duv thresholds without a secondary improvement algorithm are reported in this article.

All methods utilizing a secondary algorithm met the accuracy criteria. For each LUT type (fixed kelvin increments and fixed % increments) and each base method, a subset of methods employing a secondary Newton-Raphson algorithm is reported. This includes two Robertson-based approaches with lower resolution LUTs.

2.3 Duv Calculation

As Duv is defined as the signed distance between the (u, v) chromaticity coordinates of a light and those of the closest Planckian radiator, it can readily be calculated once the CCT has been estimated using any of the methods. This is done by calculating a Planckian spectrum with the estimated color temperature, calculating its chromaticity coordinates, and then determining the distance to the chromaticity coordinates of the test source. However, this additional step increases the calculation time, and it can be avoided while incurring only minor additional errors by calculating Duv directly from some of the intermediate quantities calculated while determining the CCT. This is already done in the Ohno base approach, and it can also be done for the other base approaches or after the Newton-Raphson algorithm has been applied.

Duv for the Ohno base approach was determined using equations 8 and 11 in Ohno (2014) for the triangular and parabolic solutions, respectively. For the Robertson base approach, Duv was determined by first estimating the (*u*, *v*) chromaticity coordinates of the closest Planckian using equation 5b in Robertson (1968), but with the reciprocal color temperature, *1/T* (which is assumed by Robertson to be a linear function of the distance along the arc of the Planckian locus between T_j and T_{j+1}), replaced by the *u* and *v* chromaticity coordinates corresponding to the various color temperatures in the equation. For the Fibonacci base approach, the (*u*, *v*) chromaticity coordinates of the closest Planckian in the LUT were used, after which the distance to the source chromaticity was calculated.

When the Newton-Raphson secondary algorithm was applied, the (*u*, *v*) chromaticity coordinates were estimated by using those of the Planckian spectrum calculated in the last iteration (see step 2 of the proposed method in Section 3 in Li et al. (Li et al. 2016)). While these do not always correspond to those of the final estimated CCT (step 11 in Li et al.) the incurred error on the Duv was usually found to be substantially smaller than the required Duv tolerance when the estimated CCT was sufficiently accurate. If the color temperature correction step size *DT* (step 10 in Li et al.) was larger than 1 K in the final iteration, the (*u*, *v*) chromaticity coordinates were calculated directly from a Planckian radiator corresponding to the final estimated CCT; this resulted in an additional increase in computation time.

Except for the Ohno base approach, which has its own technique, after the Duv absolute distance was determined for all methods, the sign of the Duv was determined by identifying the positive angle (between 0° and 360°) of the iso-temperature line passing through the source and Planckian chromaticity coordinates. If the angle was larger than 180°, a value of 360° was subtracted from the angle. Finally, the sign of the final angle was assigned to the absolute Duv value.

The code in the cct.py module in the Luxpy Python package provides more details on the calculation of the Duv (and CCT) for the various methods (Smet 2020).

2.4 Calculation Speed Considerations

Calculation speed can be an important consideration, especially when many SPDs are calculated for spectral optimizations; thus, it is an aspect of performance for comparison between methods. Calculation speed is dependent on several factors beyond the specifics of the calculation method. It can vary with the programming language used for the code (e.g., Python, C, etc.), the specific implementation of the method given a specific language (e.g., programming efficiency, batch processing, *etc*.), the computing resources (e.g., processor, memory), and the total number of conversions to be performed at once. Given the breadth of these variables, it is impossible to comprehensively sample each factor, but several options were considered.

An extensive investigation was first carried out with coding in Python (version 3.7.11), with heavy reliance on Numpy (version 1.21.5) (Harris et al. 2020) and vectorization. The analysis was performed in a Jupyter Notebook (version 6.4.6, core version: 4.9.1) running in Chrome (version 98.0.4758.82) on a 64-bit Windows 10 operating system on a HP[a] laptop with an Intel® Core™ i7-8665 CPU @ 1.9 GHz. For each method considered, calculation speeds were determined for 1, 10, 100, 1000, 10,000 and 30,000 conversions. Each test was performed 20 times with chromaticity coordinates to be converted to CCT

[a] The company and product names are given in this paper for technical information only to assist in understanding the results presented in this paper. They do not represent endorsement of this product or this manufacturer by the National Institute of Standards and Technology nor by any of authors' organizations.

and Duv randomly chosen from the 1,270-coordinate reference set—the process of choosing chromaticity coordinates was not included in the time calculation. The randomly chosen chromaticity coordinates were held constant across all methods for an unbiased comparison. Calculations in Python were performed with and without batch processing with a batch size of 25. For iterative methods, the CCT tolerance was set to 0.1 K, and the maximum number of iterations was set to 100, although this was never found to be a limiting factor. In all cases, LUTs were precalculated and stored in memory, so that computing the LUT is not part of the reported calculation time.

In Python, the number of calculations performed can influence the time per calculation due to how memory is allocated and freed during the calculation. More calculations can increase memory pressure on the chase and page table, reducing speed. While Python is widely used—and thus the calculation time in Python is important to understand—a clearer test of the pure computational efficiency of the methods was attained with a high-performance parallel programming implementation in C. Specific candidate methods were tested in C (abbreviations as noted in Table 1): R[1%,NO], O[1 K,NO] and F[0.2K,NO], R[1%,NR], and the original Newton-Raphson method proposed by Li et al. (Li et al. 2016) For these calculations, precise time per calculation was determined by averaging the difference between 20 pairs of 2 billion and 1 billion calculations—this makes a total calculation on the order of a minute. Loading of the LUT and any other auxiliary data was timed separately as the difference between 30 million and 15 million loads. The code was executed on a Linux-based computer with a 2.8 GHz AMD EPYC 7282 processor. The maximum errors for CCT and Duv matched those determined in Python, as error is independent of the programming language. In this implementation, time per CCT and Duv calculation is highly consistent. The loading time showed more variability, because it can be influenced by other system noise coming from the operating system that cannot be controlled.

3 Results

3.1 Accuracy

Figure 2 shows boxplots of the CCT and Duv errors for each of the methods considered in the final set. The 40 methods that are reported here were selected because they met the specified accuracies of ±0.1 K and ±0.0001 units for CCT and Duv, respectively, for all 1,270 reference coordinates. Only a small number of the methods initially considered could achieve this performance—in particular the 0.1 K CCT threshold—without the secondary Newton-Raphson algorithm. Of the 12 methods without a secondary algorithm (Figure 2, top), some had accuracy greater than deemed necessary (i.e., the LUT had unnecessarily high resolution). A 1% increment LUT for the Robertson base method resulted in error closer to the threshold and reduced calculation time (as subsequently discussed). At an equal LUT resolution and without a secondary algorithm, the Robertson base approach produced lower maximum errors than Ohno or Fibonacci approaches. When a secondary Newton-Raphson algorithm was included (Figure 2, bottom), the accuracy improved by approximately four orders of magnitude, even while the initial LUT increments were sometimes greater.

Figure 3a shows the maximum CCT error as a function of CCT for the methods that did not include a secondary Newton-Raphson algorithm; Figure 3b shows the same for methods that did. Note that the datapoints at each nominal CCT value represent the maximum error found within the interval in the reference set with that CCT as its center. In Figure 3a, methods with a fixed LUT increment generally showed greater maximum error at lower CCTs, whereas methods with a percentage increment in the LUT generally showed greater maximum error at higher CCTs. This is logical when considering the

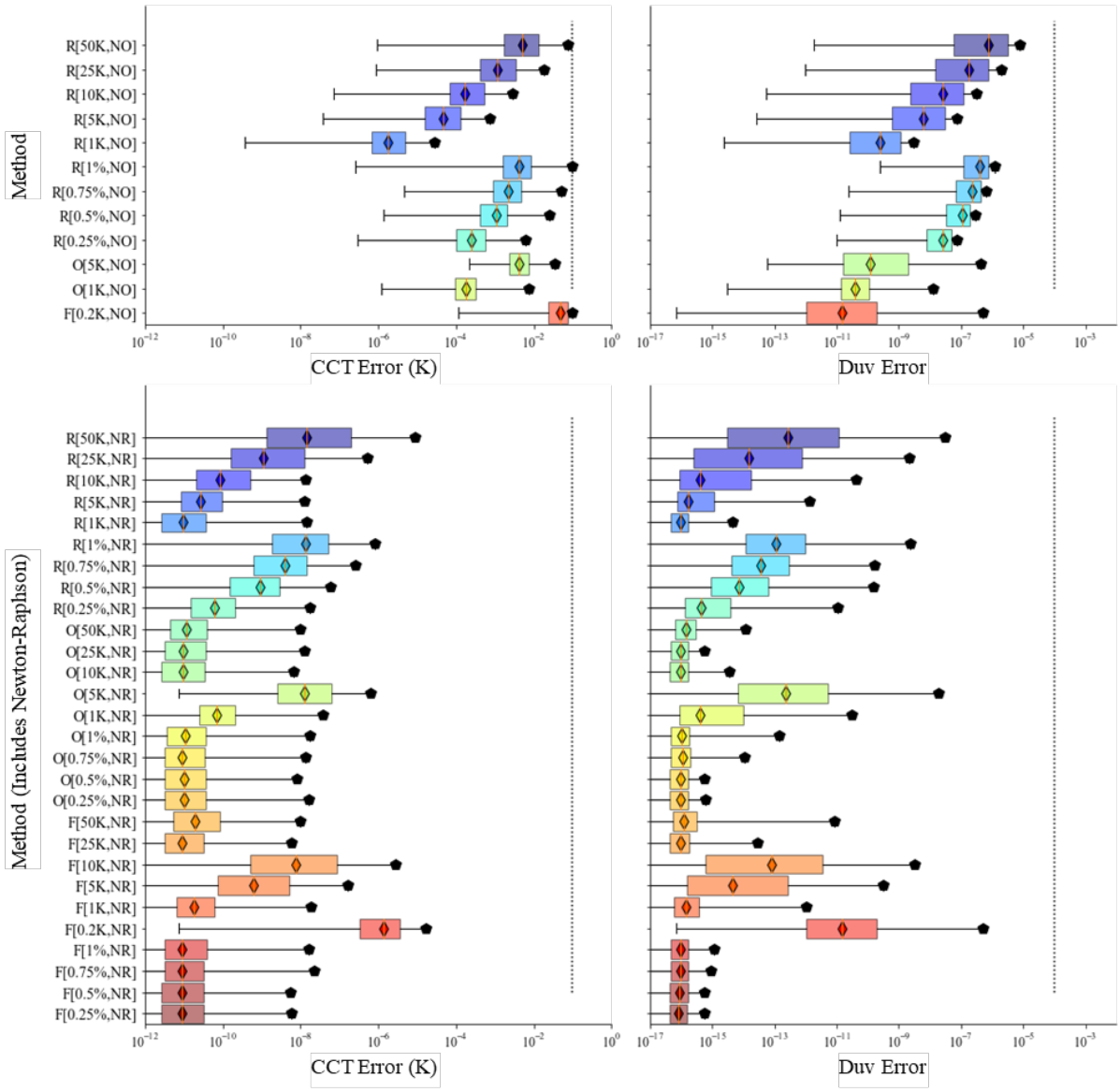

Figure 2. Boxplots of CCT error and Duv error for each method considered in the final analysis (the dashed line indicates the 0.1 K CCT and 0.0001 Duv error thresholds). The whiskers extend from the minimum to maximum value. The latter has also been highlighted with a black pentacle. The median has been indicated with an orange line and a diamond marker. Top: Base approaches. Bottom: Base approaches supplemented with a secondary algorithm; in these charts the minimums are typically not shown. Abbreviations are identified in Table 1.

variable LUT increments in kelvin when percentage increments are used. When the secondary Newton-Raphson algorithm was applied, the maximum CCT error was almost always highest at lower CCTs. In general, there was no apparent pattern to the maximum CCT error based on Duv (not shown).

While always much smaller than deemed necessary, the maximum Duv error was consistently highest at lower CCTs (under 10,000 K) and followed a more regular pattern across all methods. Without a secondary algorithm, maximum Duv error was generally constant across Duv—errors were slightly lower

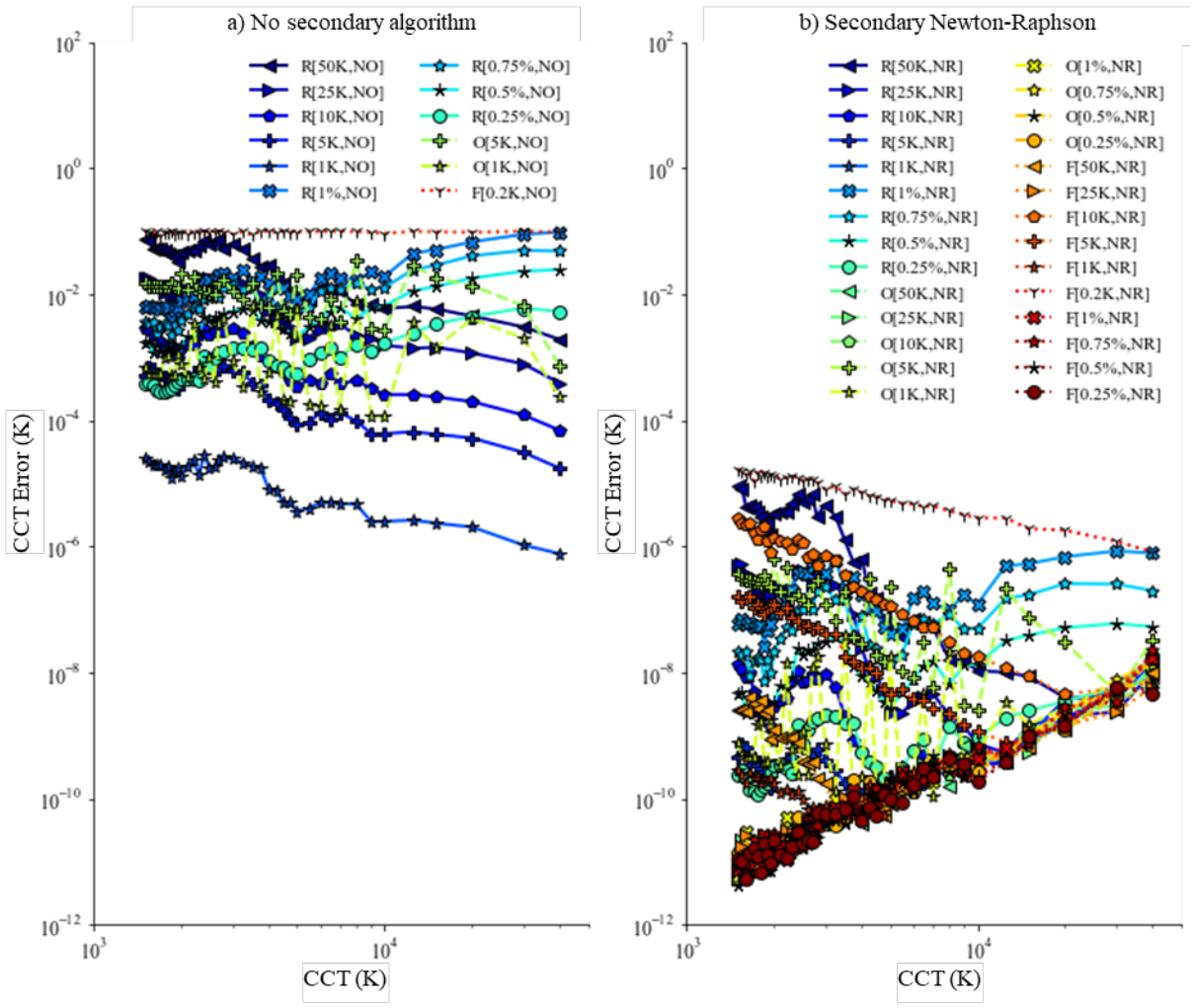

Figure 3. Maximum CCT error as a function of nominal CCT.

at higher Duv values, but the maximum Duv is bounded by the spectrum locus at lower CCTs, where error was generally higher.

3.2 Speed

Figure 4 shows boxplots of time to complete a single (*u*, *v*) to (CCT, Duv) conversion and 1,000 conversions for each of the 40 methods in Python. The variation arises from the 20 successive trials of each method making the conversion, and principally results from fluctuations in computing performance. Some variation may result from Python's global interpreter lock, which does not allow the user to dictate how memory is used. As expected, methods with a higher-resolution LUT generally had longer calculation times. Except for those with the highest resolution, the calculation time per CCT was less than a millisecond.

Adding a secondary Newton-Raphson algorithm to an existing method with the same LUT will increase the median calculation time—with the previously reported benefit of approximately four orders of

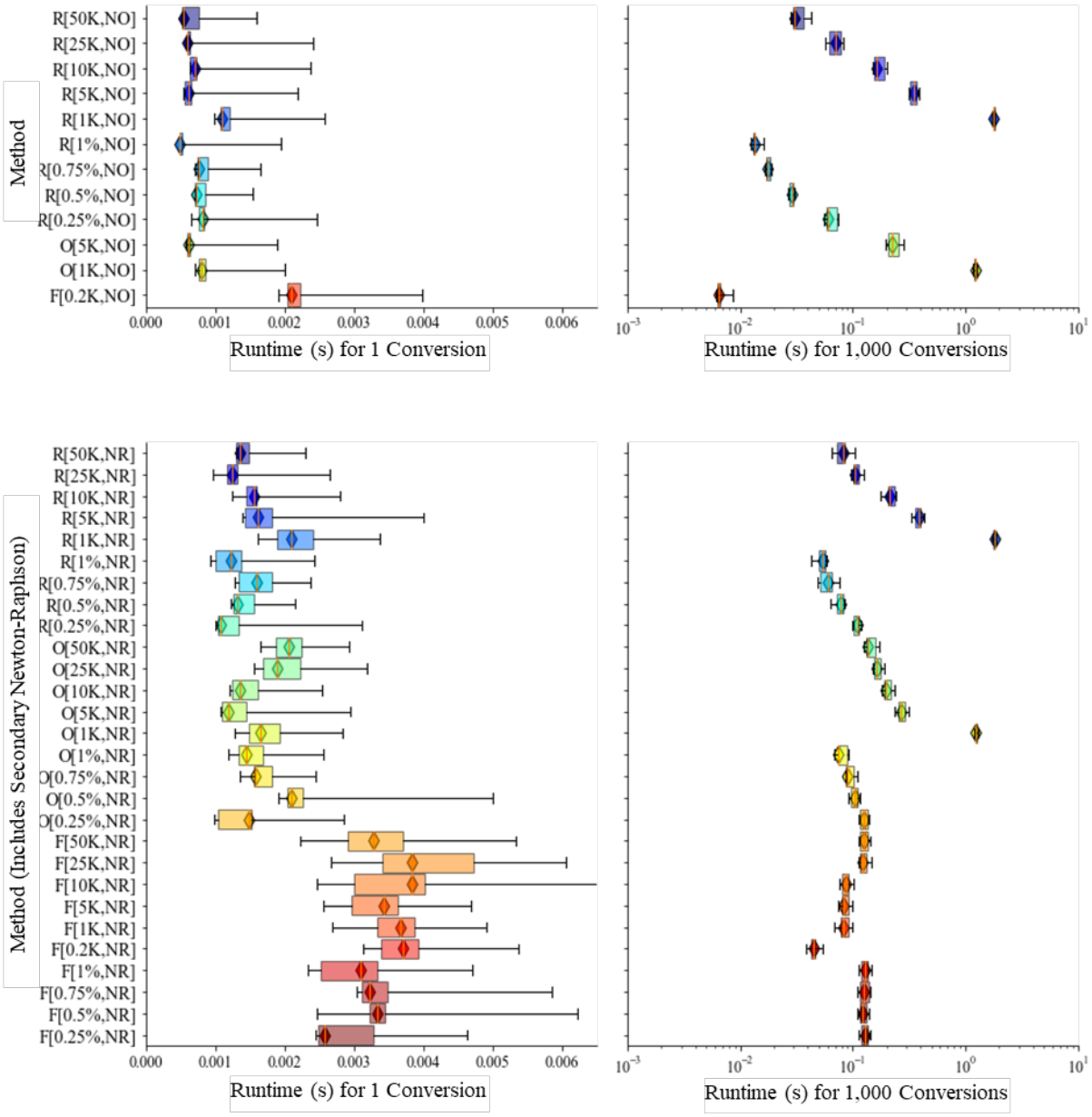

Figure 4. Left column: Boxplots of time for a single conversion for each of the 40 methods in Python. Right columns: Boxplots of time for 1,000 conversions for each of the 40 methods in Python (note logarithmic scale). The whiskers extend from the minimum to maximum value. The median has been highlighted with an orange line and a diamond marker.

magnitude improvement in accuracy. Many LUT resolutions—as coarse as 1000 K and 15%—were evaluated for methods using a Newton-Raphson algorithm. Adding a secondary algorithm, such as NR, always increased calculation time for all the base methods considered in this study. Said another way, all the fastest methods in this study did not include a secondary algorithm.

Figure 5 explores the relationship between calculation time and number of conversions, showing median calculation time in Python as a function of number of conversions, all relative to the fastest (or

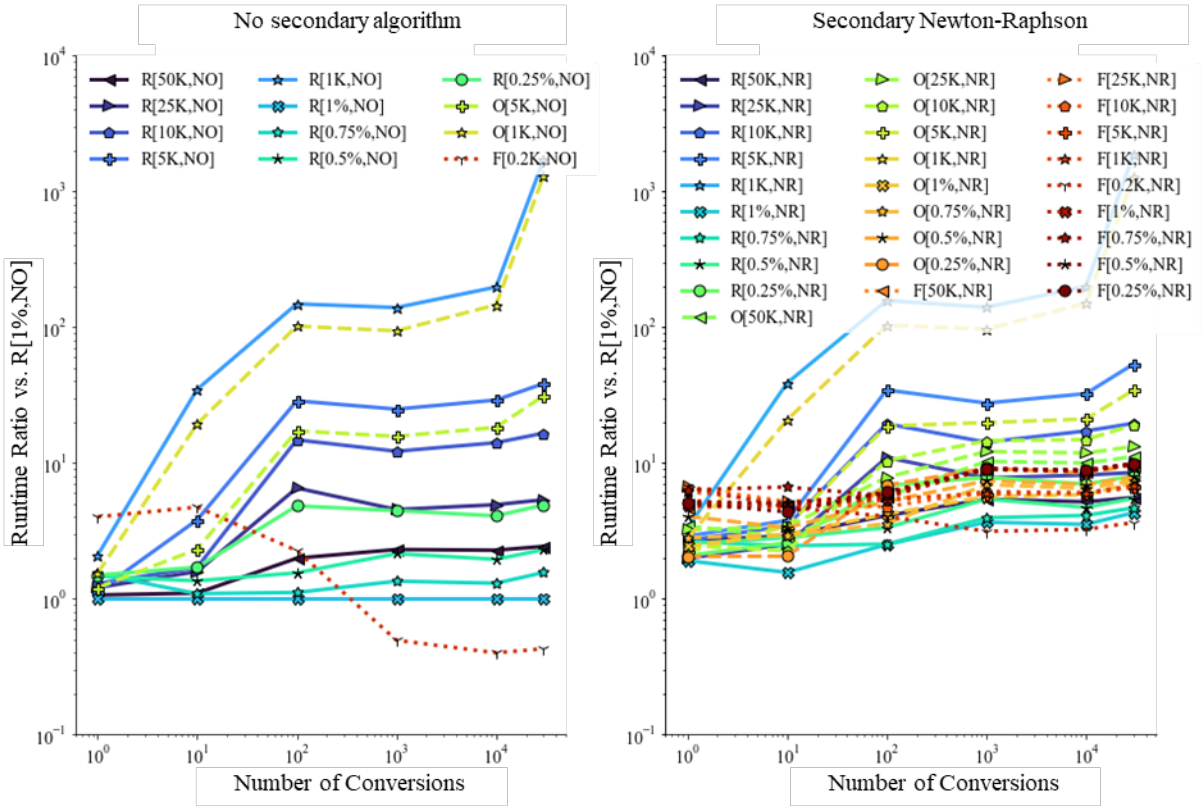

Figure 5. Relative calculation time compared to the R[1%,NO] method, at each tested number of conversions.

close to) single-conversion method, R[1%,NO], at the tested number of conversions. Values greater than 1 reveal a relatively slower calculation compared to the reference method, whereas values less than 1 reveal a relatively faster calculation. Figure 5 shows the point around 400 conversions where the F[0.2K,NO] became slightly faster than the R[1%,NO]. This result did not change substantially when the secondary Newton-Raphson was applied, although the speed advantage of the R[1%,NO] diminished somewhat.

The results from the computations in C generally agree with those from Python, but as previously noted, they provide a better raw indication of each method's efficiency and do not change with the number of calculations performed. Table 2 provides key results for selected candidate methods—not all methods were programmed and timed in C. The fastest method that reached the desired accuracy threshold of a maximum error of 0.1 K was the R[1%,NO] method. This method had about a 40% faster mean time per calculation than the Ohno-based method with the greatest maximum error under the limit, O[5K,NO]. With the exact same 1%-increment LUT, the Robertson interpolation calculations are about 10% more efficient than the Ohno interpolation calculations, and about 26 times more accurate. Lastly, R[1%,NO] was also about 73 times faster per CCT than the same method with a secondary Newton-Raphson R[1%,NR].

*Table 2. Mean calculation times for selected methods programmed in C. Each time is the average of billions of calculations. *Uses original 31-row LUT proposed by Robertson (1968). **Does not meet accuracy criterion for all points in reference set—included for direct comparison of mean time per calculation.*

Method	Mean Time per Calculation (ns)
R[1%,NO]	80
O[1%,NO]**	89
O[5K,NO]	132
O[1K,NO]	165
F[0.2,NO]	143
R[Orig,NR]*	9,600
R[1%,NR]	5,900

4 Discussion

Consistent with prior research, preliminary analyses found that the original Robertson (max CCT error = 151.7 K) and Ohno (1% LUT max CCT error = 2.6 K; 0.25% LUT max CCT error = 0.4 K[b]) methods with their native LUTs failed to achieve the *a priori* tolerance thresholds of 0.1 K for CCT over the considered range of 1500 K to 40,000 K and Duv of -0.05 to 0.05, using the 1,270 reference coordinates. This is principally because for their corresponding base method, their LUTs have (increasing) kelvin increments that are too large for some CCTs. Increasing the resolution of the LUTs to a maximum increment of 50 K or 1% (i.e., R[50 K,NO] or R[1%,NO]) for the Robertson method or 5 K (i.e., O[5K,NO]) for the Ohno method allows each to meet the maximum error tolerance. The Fibonacci method required an increment of 0.2 K or smaller. Notably, the interpolation method of Robertson can be used with larger LUT increments than the Ohno method while still meeting the accuracy threshold. An alternative to substantially reducing the LUT increment is applying a secondary Newton-Raphson algorithm.

In terms of speed, for LUTs from 1000 K to 40,000 K, the Robertson approach with a 1% LUT and no secondary algorithm, R[1%,NO], was generally the fastest method per CCT and Duv calculation that met the accuracy criteria. This was true in both C and Python. For larger number of conversions the F[0.2K,NO] method was the fastest in Python due to the way that Python processes data. In the C implementation, CCT calculation speed was independent of the number of conversions performed, and the R[1%,NO] was 40% faster than the Ohno-based method with the lowest-resolution LUT (5 K) from the tested set that met the CCT accuracy criterion. This is primarily because of the higher resolution of the LUT needed for Ohno to meet the accuracy threshold, but also because the Robertson interpolation algorithm is slightly more efficient.

Iterative methods based on cascading LUTs or adding a Newton-Raphson algorithm to the initial estimate met the tolerance threshold, but also added considerable calculation time and complexity. The Newton-Raphson algorithm, regardless of initial LUT and interpolation method, decisively outperformed the cascading LUT algorithm in terms of calculation speed (details not reported). With the Newton-

[b] The maximum error for the Ohno method with a 0.25% LUT can be reduced to 0.12 K by optimizing the CCT-based correction factor and parabolic-to-triangular crossover point for that specific LUT.

Raphson approach, there was a tradeoff between the resolution of the LUT used for the initial estimate—and thus the accuracy of the initial estimate—and the number of iterations needed to reach the tolerance threshold. Decreasing the increment of the LUT to rely more on the Newton-Raphson algorithm did not improve calculation time compared to just using a higher-resolution LUT with no secondary algorithm—additional iterations in the method add more calculation time than is saved with a faster LUT search. Any method using a secondary Newton-Raphson algorithm was more than 70 times slower for performing the CCT and Duv calculations in C—but offered considerably improved accuracy. In Python, methods adopting a Newton-Raphson were minimally 1.5 times slower than R[1%,NO], but could be as much as 1,000 times slower depending on the method, the LUT size and the number of conversions being performed. Adding a Newton-Raphson in Python to R[1%,NO] was 4 times slower for 30,000 conversions.

It is evident that there are multiple possible ways to achieve the desired level of accuracy for CCT and Duv calculations: very-high-resolution tables with no interpolation (e.g., efficiently searched with the Fibonacci approach), moderately high-resolution tables with an interpolation algorithm (e.g., Robertson and Ohno approaches), or lower resolution tables with an interpolation algorithm for an initial estimate and then iterative methods (e.g., Newton-Raphson). The most efficient approach depends on the particulars of the implementation, including the programming language, number of calculations performed, and the components included in the time (i.e., with or without table loading).

The Robertson method with a 1% LUT offers several advantages: with or without a secondary Newton-Raphson algorithm, it is among the most efficient methods regardless of implementation or number of calculations (where relevant). Furthermore, its accuracy can be easily increased by several orders of magnitude by “turning on” the Newton-Raphson secondary algorithm. A single, simple base method, with relatively low storage LUT, can provide accurate (without a secondary Newton-Raphson) to extremely accurate (with a secondary Newton-Raphson) CCT and Duv calculations at speeds which are the fastest or close to the fastest for any number of conversions. For these reasons, if one reference method is to be chosen, we recommend the Robertson method with a 1% LUT approach (denoted [R[1%,NO] in this article) for consideration by lighting standards organizations, which currently do not recommend any single method for calculating CCT and Duv.

4.1 Limitations

This work considered a broad range of combinations of methods that were candidates to deliver a fast calculation while meeting *a priori* targets for CCT and Duv accuracy. It is possible that LUT increments could be further optimized to deliver gains in calculation speed without the CCT and Duv errors exceeding their tolerances.

Computation time is dependent on many factors, including the computer performance the calculations and the implementation of the method in a specific coding language. We compared performance using two implementations. It is likely that additionally implementations could produce slightly different results, but the general trends are not expected to change.

Achieving the specified accuracies requires that computations are performed—and intermediate data stored—with sufficient precision. Standard double precision was used for this analysis.

5 Conclusion

This work explored various methods for computing CCT and Duv considering accuracy, speed, complexity, and ease of use. The goal was to recommend a method for standardization by a consensus body that could achieve a maximum CCT error of 0.1 K and a maximum Duv error of 0.0001 in the CCT range of 1500 K to 40,000 K and Duv range of 0.05 to -0.05, with a fast calculation time, reasonable data storage needs, and that is practical to implement. The latter three considerations go beyond traditional evaluations, which have focused on demonstrating improved accuracy in estimating CCT. After an extensive exploration of several hundred methods, 40 were thoroughly evaluated, with independent implementation in Python. A subset of the 40 methods evaluated in Python was then implemented in C. Accuracy was examined using a set of 1,270 reference coordinates. Speed in Python was determined by converting randomly selected 1, 10, 100, 1,000, 10,000, and 30,000 (u, v) coordinates to CCT and Duv values, with 20 replications of each. Speed in C was determined by averaging the difference between 2 billion and 1 billion calculations, repeated 20 times.

Existing methods from Robertson and Ohno with LUTs in their original publication did not meet the desired accuracy maxima. Finer resolution LUTs for Robertson (25 K, 10 K, 5 K, 1 K, 0.2 K, 1%, 0.75%, 0.5%, 0.25%) and for Ohno (5 K, 1 K, 0.2 K, and 0.2 %) were defined that could achieve the accuracy limits without any changes to the prescribed interpolation algorithms of each method. For less than around 400 conversions, the Robertson method with a 1% LUT—having 372 rows—was the fastest method meeting the accuracy thresholds in the Python implementation. The Robertson approach with a 1% LUT was always the fastest method in C. This implementation only marginally increases the initial load time compared to a coarse 30-row LUT, as used in Robertson's original work. In addition, using a Fibonacci method to search the LUT, as recently described, means it is typically as fast at calculating CCT (and Duv) values as the original, less-accurate method. For these reasons, we recommend the Robertson method with a 1% LUT for consideration and adoption by lighting standards organizations.

Funding

This work of Michael Royer was supported by the U.S. Department of Energy's Lighting R&D Program, part of the Building Technologies Office within the Office of Energy Efficiency and Renewable Energy (EERE).

Disclosure Statement

The authors report there are no competing interests to declare.

6 References

Avriel M, Wilde DJ. 1966. Optimally proof for the symmetric fibonacci search technique. Fibonacci Q J.:265–269.

Baxter D, Royer M, Smet K. 2023. Modifications of the Robertson Method for Calculating Correlated Color Temperature to Improve Accuracy and Speed. Leukos. doi:10.1080/15502724.2023.2166060.

Commission Internationale de l'Eclairage. 2011. International Lighting Vocabulary. CIE S017/E:2020. Vienna, Austria: Commission Internationale de l'Eclairage.

Commission Internationale de l'Eclairage. 2017. CIE 2017 colour fidelity index for accurate scientific use. CIE 224:2017. Vienna, Austria: Commission Internationale de l'Eclairage.

Commission Internationale de l'Eclairage. 2018. Colorimetry, 4th Edition. CIE 15:2018. Vienna, Austria: Commission Internationale de l'Eclairage.

Davis R. 1931. A correlated color temperature for illuminants. Nat Bur Stand J Res. 7:659.

Durmus D. 2022. Correlated color temperature: Use and limitations. Light Res Technol. 54(4):363–375. doi:10.1177/14771535211034330.

Gardner JL. 2000. Correlated colour temperature - uncertainty and estimation. Metrologia. 37(5):381–384. doi:10.1088/0026-1394/37/5/8.

Guo X, Houser KW. 2004. A review of colour rendering indices and their application to commercial light sources. Light Res Technol. 36(3):183–199. doi:10.1191/1365782804li112oa. http://journals.sagepub.com/doi/10.1191/1365782804li112oa.

Harding HGW. 1950. The colour temperature of light sources. Proc Phys Soc Sect B. 63(9):685–698. doi:10.1088/0370-1301/63/9/306.

Harris CR, Millman KJ, van der Walt SJ, Gommers R, Virtanen P, Cournapeau D, Wieser E, Taylor J, Berg S, Smith NJ, et al. 2020. Array programming with NumPy. Nature. 585(7825):357–362. doi:10.1038/S41586-020-2649-2.

Hernández-Andrés J, Lee RL, Romero J. 1999. Calculating correlated color temperatures across the entire gamut of daylight and skylight chromaticities. Appl Opt. 38(27):5703. doi:10.1364/ao.38.005703.

Illuminating Engineering Society. 2020a. ANSI/IES LS-1-20: Lighting Science: Nomenclature and Definitions for Illuminating Engineering. New York, NY.

Illuminating Engineering Society. 2020b. IES Method for Evaluating Light Source Color Rendition. ANSI/IES TM-30-20. New York, NY: Illuminating Engineering Society.

Judd DB. 1936. Estimation of chromaticity differences and nearest color temperature on the standard 1931 ICI colorimetric coordinate system. J Res Natl Bur Stand (1934). 17(5):771. doi:10.6028/jres.017.045.

Kelly KL. 1963. Lines of constant correlated color temperature based on MacAdam's (u, υ) uniform chromaticity transformation of the CIE diagram. JOSA. 53(8):999–1002.

Krystek M. 1985. An algorithm to calculate correlated colour temperature. Color Res Appl. 10(1):38–40.

Li C, Cui G, Melgosa M, Ruan X, Zhang Y, Ma L, Xiao K, Luo MR. 2016. Accurate method for computing

correlated color temperature. Opt Express. 24(13):14066. doi:10.1364/oe.24.014066.

Li Y, Gao C, Melgosa M, Li C. 2022. Improved Methods for Computing CCT and Duv. Leukos. Online bef. doi:10.1080/15502724.2022.2081175.

MacAdam DL. 1977. Correlated color temperature? J Opt Soc Am. 67(6):839. doi:10.1364/josa.67.000839.

McCamy CS. 1992. Correlated color temperature as an explicit function of chromaticity coordinates. Color Res Appl. 17(2):142–144. doi:10.1002/col.5080170211.

Mori L, Sugiyama H, Kambe N. 1964. An Absolute Method of Color Temperature Measurement. Acta Chromatica. 1(3):93–102.

Ohno Y. 2014. Practical use and calculation of CCT and Duv. Leukos. 10(1):47–55. doi:10.1080/15502724.2014.839020. http://dx.doi.org/10.1080/15502724.2014.839020.

Organization [ISO] International Standards. 2022. ISO/CIE DIS 11664-2.2 - Colorimetry — Part 2: CIE standard illuminants. Geneva, Switzerland. [accessed 2021 Dec 27]. https://www.iso.org/standard/77215.html.

Overholt KJ. 1973. Efficiency of the Fibonacci search method. BIT Numer Math. 13(1):92–96.

Prytkov S V., Kolyadin M V. 2021. Error estimation for the methods of correlated colour temperature calculation. Light Eng. 29(3):70–77. doi:10.33383/2021-018.

Robertson AR. 1968. Computation of Correlated Color Temperature and Distribution Temperature. J Opt Soc Am. 58(11):1528. doi:10.1364/josa.58.001528.

Schanda J, Mészáros M, Czibula G. 1978. Calculating correlated color temperature with a desktop programmable calculator. Color Res Appl. 3(2):65–69.

Smet KAG. 2020. Tutorial: The LuxPy Python Toolbox for Lighting and Color Science. LEUKOS. 16(3):179–201. doi:10.1080/15502724.2018.1518717. https://doi.org/10.1080/15502724.2018.1518717.

Wyszecki G, Stiles WS. 1982. Wyszecki, Gunter, and Walter Stanley Stiles. Color science. Vol. 8. New York: Wiley, 1982.

Xingzhong Q. 1987. Formulas for computing correlated color temperature. Color Res Appl. 12(5):285–287.

Ypma TJ. 1995. Historical development of the Newton–Raphson method. SIAM Rev. 37(4):531–551.

Zhang F. 2019. High-accuracy method for calculating correlated color temperature with a lookup table based on golden section search. Optik (Stuttg). 193(June):163018. doi:10.1016/j.ijleo.2019.163018. https://doi.org/10.1016/j.ijleo.2019.163018.

Zheleznikova OE. 2020. Selection of Optimal Method of Correlated Colour Temperature Calculation. J Mech Contin Math Sci. spl8(1):134–143. doi:10.26782/jmcms.spl.8/2020.04.00010.