

Computationally efficient algorithms for the two-dimensional Kolmogorov-Smirnov test

Raul H C Lopes, Peter R Hobson and Ivan D Reid

School of Engineering and Design, Brunel University, Uxbridge UB8 3PH, United Kingdom

E-mail: Raul.Lopes@brunel.ac.uk

Abstract. Goodness-of-fit statistics measure the compatibility of random samples against some theoretical or reference probability distribution function. The classical one-dimensional Kolmogorov-Smirnov test is a non-parametric statistic for comparing two empirical distributions which defines the largest absolute difference between the two cumulative distribution functions as a measure of disagreement. Adapting this test to more than one dimension is a challenge because there are 2^d-1 independent ways of ordering a cumulative distribution function in d dimensions. We discuss Peacock's version of the Kolmogorov-Smirnov test for two-dimensional data sets which computes the differences between cumulative distribution functions in $4n^2$ quadrants. We also examine Fasano and Franceschini's variation of Peacock's test, Cooke's algorithm for Peacock's test, and ROOT's version of the two-dimensional Kolmogorov-Smirnov test. We establish a lower-bound limit on the work for computing Peacock's test of $\Omega(n^2 \lg n)$, introducing optimal algorithms for both this and Fasano and Franceschini's test, and show that Cooke's algorithm is not a faithful implementation of Peacock's test. We also discuss and evaluate parallel algorithms for Peacock's test.

1. Introduction

Goodness-of-fit statistics measure the compatibility of random samples against some theoretical probability distribution function. In general, given two independent stochastic variables X and Y whose cumulative distribution functions (CDFs) F and G are unknown, the classical two-sample problem consists of testing the null hypothesis

$$H_0 : F(x) = G(x), \text{ for every } x \in R^d$$

against the general alternative

$$H_1 : F(x) \neq G(x), \text{ for some } x \in R^d$$

This kind of problem could arise in a context where, given an observed sample X_1, \dots, X_n and a reference sample Y_1, \dots, Y_m , one must determine whether they come from the same distribution function. The nature of the sets is, however, important in defining the kind of test available. In particular, an important consideration is whether the data are available as discrete points or have been binned into histograms. A well accepted test for binned distributions is based on the χ^2 statistic[1]. Continuous data can always be binned by grouping the events into ranges, but this usually comes at the price of losing information.

The classical test for one-dimensional continuous data is the Kolmogorov-Smirnov (KS) test[2]. It is a non-parametric statistic for comparing two empirical distributions which defines the largest absolute difference between the two cumulative distribution functions as a measure of disagreement. Adapting goodness-of-fit tests to multi-dimensional space is in general a challenge. Tests based on binning face the hurdle of “the curse of dimensionality”: a high dimensional space is mostly empty and binning tests can only start to be effective when the data sets are very large[3]. On the other hand, adapting the KS test demands the definition of a probability function that is independent of the direction of ordering, which does not seem to be possible given that there are $2^d - 1$ ways of defining a CDF in a d -dimensional space.

However, there are many applications in experimental physics where comparison of two-dimensional data sets is important. For example, monitoring the performance of high-energy physics detectors such as the Compact Muon Solenoid[4] (CMS) at the Large Hadron Collider will involve periodic comparisons of collected data to reference histograms to uncover as soon as possible any changes in operating conditions; the sheer number of detector elements to be monitored necessitates that this process be automated as much as possible.

As an example of such histograms we have produced data from the reconstruction of the simulated tracks of muon pairs, originating from Z -particle decay, within the CMS Silicon Tracker sub-detector. The data, obtained using the CMSSW software framework[4], are given as histograms in Fig. 1, showing the relationship between the reduced χ^2 of the track fit and the track pseudorapidity η ($\eta = -\ln \tan \frac{\theta}{2}$, where θ is the angle between the track and the proton beam axis). The first histogram gives results for perfect detector alignment while the second histogram was obtained after introducing small errors, representative of probable initial errors[5], to individual detector modules. The data are binned at 20x20 resolution; the blockiness is due to dithering each bin to fill its area proportionally to its contents. It is noticeable that χ^2 is generally higher in the second histogram around $\eta = \pm 1.5$, where tracks pass through the transition between cylindrical “barrel” detectors and circular “end caps” [4]. Files of the individual data points were retained, and subsets of the data used in the comparisons in this work, which extends upon results reported earlier[6].

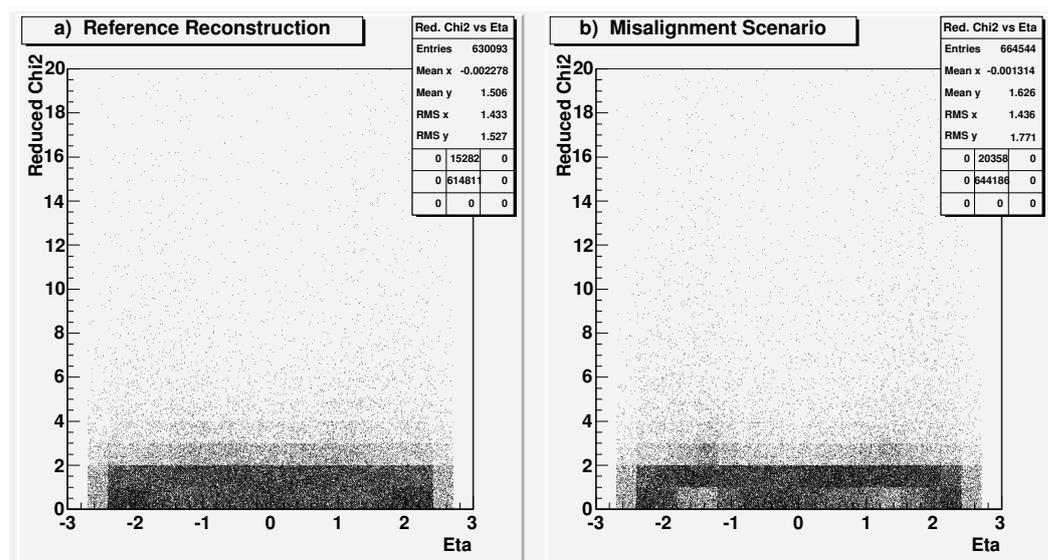


Figure 1. 2D data used in this study; reduced χ^2 of track fits *vs.* pseudorapidity η for reconstructed muon tracks in the CMS Silicon Tracker. a) Ideal detector geometry. b) small “short-term scenario” misalignments of the detector modules. The histograms use 20x20 binning.

2. Peacock's variation on the KS test

Table 1. Results and timings (in seconds) of implementations of Peacock's KS test and the Fasano and Franceschini algorithm for comparisons of different-sized subsamples of the aligned and misaligned track data. *Programmes in C on Ubuntu Linux; 2.0 GHz AMD Athlon XP.*

Sample Size	Peacock				Fasano and Franceschini			
	Brute Force Distance	Brute Force t (s)	Range-Counting Distance	Range-Counting t (s)	Brute Force Distance	Brute Force t (s)	Range-Counting Distance	Range-Counting t (s)
1024	0.096680	110	0.096680	35.5	0.094727	0.15	0.094727	0.04
2048	0.084961	969	0.084961	180	0.081055	0.60	0.081055	0.13
3072	0.072591	7357	0.072591	502	0.069824	1.38	0.069824	0.24
4096	0.075195	19143	0.075195	1130	0.073364	2.46	0.073364	0.38
5120	0.078320	37591	0.078320	1970	0.075684	3.84	0.075684	0.52
6144	0.084147	64887	0.084147	2613	0.082601	5.53	0.082601	0.67
7168	0.086356	103208	0.086356	3758	0.085658	8.51	0.085658	0.83
8192	0.083374	153833	0.083374	5144	0.082520	11.7	0.082520	1.11
65536	-	-	-	-	0.075539	840	0.075539	20.1
131072	-	-	-	-	0.074738	1318	0.074715	52.4

The KS test is applicable to continuous, unbinned, one-dimensional data samples and assumes that a list of data points can be easily converted to a cumulative distribution function. To extend this to two dimensions, given the challenge noted above, Peacock[7] introduced the idea of making the statistic independent of any particular ordering by finding the largest difference between the cumulative distribution functions under any possible ordering. Given n points in a two-dimensional space, this requires calculating the cumulative distribution functions in the $4n^2$ quadrants of the plane defined by all pairs (X_i, Y_j) , X_i and Y_j being coordinates of any pair of points in the given samples. The test can be performed by a brute force algorithm which sweeps through each quadrant for every point in a sample, deciding whether the point is in it. This requires n steps, one for each point, with complexity $\Theta(n^2)$, thus giving a final complexity of $\Theta(n^3)$.

This time can be improved if, instead of deciding whether each individual point is contained in each quadrant, we apply an efficient algorithm to query the number of points in each quadrant. Given that Peacock's test defines $4n^2$ quadrants, the time of the new algorithm will be $O(n^2q)$, where q is the time spent in each query. One suitable query method uses the range-counting tree, a balanced binary tree where each node splits the points under it by the median of their x coordinate. In addition, using a cascading fractional technique developed in [8], each node indexes both nodes to its left and right using an ordered sequence of the y coordinate of the points under it. A set of n points in two dimensions can be indexed by a range-counting tree in $O(n \lg n)$ [9]. To use such a range-counting tree, a two-sided query is started using a binary search in the root to decide the largest possible interval of y coordinates and a comparison of the x coordinate in the query with the x median in the root of the tree to direct a recursive search to either the left or the right sub-tree. Any two-sided range-counting query can be answered in $O(n \lg n)$ time. The accumulation of the largest difference between the cumulative distribution functions in each quadrant defined by Peacock's test can be performed by one range-counting query, and the whole test can be computed in $O(n^2 \lg n)$.

The lower-bound for finding the maximum of $O(n^2)$ quantities is $\Omega(n^2)$ [10]. In addition, the lower-bound for performing Peacock's test on n points is the accumulation of the $4n^2$ differences

of cumulative distribution functions which requires $\Omega(n^2 \lg n)$, because it needs to locate each point in the quadrants and the lower-bound for this is $\Omega(\lg n)$.

Peacock's test is very demanding. Performing the test on 2^{18} points with the brute-force algorithm running on a 2GHz processor would require several years. Even the range-counting tree based algorithm would demand days to perform the test on such a sample, as shown by Table 1 which gives the results and running times, in seconds, obtained by using the two techniques to compare different-sized subsamples of the aligned and misaligned track data.

3. Fasano and Franceschini's test

Fasano and Franceschini introduced a variation[11] on Peacock's test that evaluates the model distributions only in the quadrants centred on each point of the given samples – a sample with n points defines $4n$ quadrants. A brute force algorithm performs a sweep through all quadrants for each given point, to decide whether the point must be counted in it. This algorithm computes in $\Theta(n^2)$ and is presented, for example, in [12].

An algorithm based on a range-counting tree can index the n points in $O(n \lg n)$ time. Subsequently $4n$ two-sided range queries of $O(\lg n)$ each can be used to compute cumulative distribution functions differences in all quadrants. The lower-bound for computing the Fasano and Franceschini test is the lower-bound for sorting n points in a two-dimensional plane, i.e. $O(n \lg n)$ [13]. As such, the algorithm based on a range-counting tree is optimal.

Table 1 also shows results and running times, in seconds, for both these implementations of Fasano and Franceschini's test. The Table indicates that all running times follow the predicted scaling with sample size, and that for sets with up to a thousand (2^{10}) points any of the algorithms demands only a few minutes of computation. For more than one million points (2^{20}), however, only the Fasano and Franceschini test implemented on top of a range-counting tree can yield times in the range of minutes.

4. Cooke's test

Cooke has presented an efficient method which is claimed to calculate Peacock's test[14]. The algorithm is described and implementations for it given in both Python and C. A parallel implementation of Cooke's algorithm is discussed by Chan[15].

Cooke's algorithm is evaluated here because it is the fastest of all tests based on the 1-D KS test, even though, as shown below, the claim that it runs in $O(n \lg n)$ is optimistic. The algorithm uses two binary trees each containing all points from both samples, with each point tagged to identify its source. The trees are ordered in x and in the first tree points are inserted in increasing order of the y coordinate. As a result, points with lower y coordinate will be allocated next to the root. By moving down the tree from leaves to root, the algorithm performs a sweep from top to bottom in all quadrants, each node in the tree defining the centre of a quadrant. The second tree inverts the order of insertion of points, locating points with higher y coordinates next to the root to produce a sweep from bottom to top in the quadrants as the tree is swept from leaves to root. The number of points in each quadrant is updated during the sweeps, by counting the number of points in each sub-tree, and updating the maximum difference.

Cooke's unproven assumptions about the algorithm are that the dominating time is in the construction of the tree, which is claimed to scale as $O(n \lg n)$, and that by sweeping all quadrants defined by the nodes from top to bottom and reverse the computation is done over all $4n^2$ quadrants defined by Peacock's algorithm.

The first assumption is clearly false. The algorithm uses an unbalanced binary tree and the upper-bound to build such a tree is $O(n^2)$ [16, 17]. The upper limit to constructing a balanced binary tree is $O(n \lg n)$, but since the algorithm depends on ordering the tree level by y coordinate a balanced tree cannot be used as this would disturb the ordering in the levels. However, the algorithm is still efficient as the *average* time to build an unbalanced tree is $O(n \lg n)$.

The second assumption is also false. A sample with n repeated points defines four quadrants in Peacock's test. In Cooke's algorithm, each point constitutes a node in the binary tree and a reference for a quadrant: n different points with the same coordinates implies n different nodes in the binary tree, yielding a quadratic time in the construction because the tree will be totally unbalanced, and also a statistic that is different from the one produced by Peacock's test.

Table 2 compares distances and running times for Cooke's and Peacock's tests. The sub-samples of the χ^2 versus η data used in these comparisons are:

- aeta.8: 256 points from the aligned data
- meta.9: 512 points from the misaligned data
- r7m4: artificial sample – 128 repetitions of a sample of 16 points from the misaligned data
- r8m4: artificial sample – 256 repetitions of a sample of 16 points from the misaligned data
- r9m4: artificial sample – 512 repetitions of a sample of 16 points from the misaligned data
- aeta.13: 8192 points from the aligned data
- meta.13: 8192 points from the misaligned data

It is interesting to notice that when the samples contain repetitions Cooke's running times quadruple for a doubling of the input size. As well, the last two lines of the table show that the test was more than 30 times faster in the absence of repetitions for inputs containing 8192 points each. However, a more pertinent observation is that for comparisons of sub-samples with themselves Cooke's test returns a finite distance, implying that there is a difference between the distributions, while Peacock's test returns the expected zero maximum distance. This is because the binary trees contain data from both distributions, so a self-comparison leads to repeated points in the tree, with the detrimental effects noted above.

Table 2. A comparison of Cooke's and Peacock's tests on data sub-samples. *Programmes in C on Ubuntu Linux; 2.0 GHz AMD Athlon XP.*

Samples	Size	Cooke's Test		Peacock's Test	
		Distance	Time (s)	Distance	Time (s)
aeta.8 vs. aeta.8	256	0.031250	0.00	0.000000	0.06
meta.9 vs. meta.9	512	0.015625	0.00	0.000000	0.29
r7m4 vs. r7m4	2048	0.066406	0.43	0.000000	12.4
r8m4 vs. r8m4	4096	0.077637	1.75	0.000000	1023
r9m4 vs. r9m4	8192	0.061890	5.41	0.000000	5259
aeta.13 vs. meta.13	8192	0.0833374	0.17	0.0833374	5259

5. ROOT's KS test

The data analysis package ROOT[18] is widely used within the physics community. It is particularly oriented towards histogrammed data, and provides KS comparisons within its one- and two-dimensional histogram classes. While these cannot give a true metric like Peacock's test, they can nevertheless be useful.

The calculation of a pseudo-KS statistic is trivial in the 1D histogram case, but in the 2D case the implementation runs up against the problem of how to define a suitable ordering to calculate the CDF. The chosen solution is to calculate two CDFs for each histogram, running over the bins in row-major and column-major order respectively, then calculating the KS probability from the average of the maximum distance between the two sets of CDFs. An example of these pseudo-CDFs is illustrated in Fig. 2, for a comparison between 8192-track samples from the aligned and misaligned data (aeta13 and meta13 above) binned at 50 x 50.

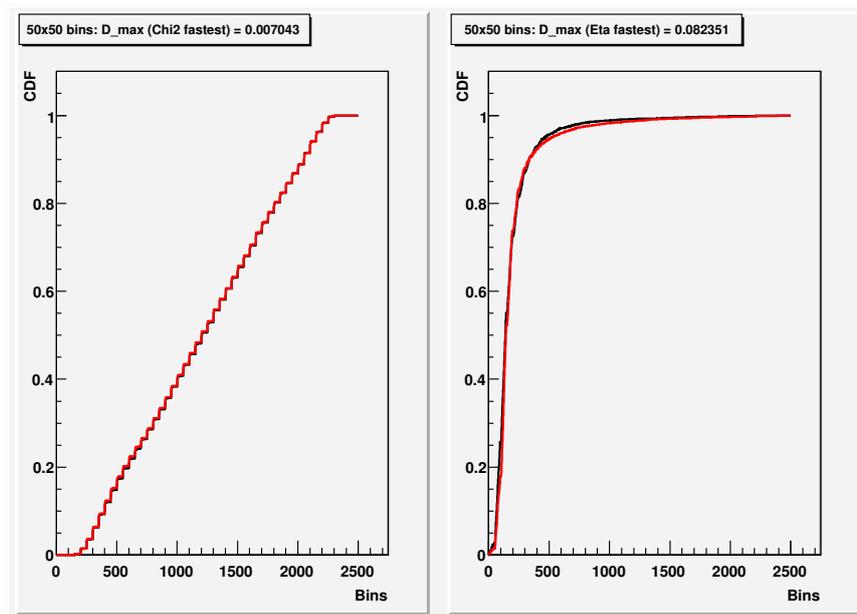


Figure 2. The two pairs of pseudo-CDFs accumulated by ROOT's 2D KS test during a comparison of two 8192-track histograms binned at 50 x 50.

Table 3. The ROOT 2D KS comparison test applied to two 8192-track data subsamples. Shown are the maximum differences D_{\max} between the pseudo-CDFs obtained from the two different orderings of the histogram bins – all tests were consistent with the samples' being from different populations. Discrete 1D KS tests, using RPy, on the χ^2 and η data separately are included for comparison. Results from ROOT's χ^2 comparison test are also given. *ROOT 5.13 and RPy libraries called from python under Scientific Linux CERN 3.08; 2.8 GHz Intel Pentium D.*

Histogram Size	$D_{\max}(\chi^2)$	$D_{\max}(\eta)$	χ^2 Test	Time (s)
25 x 25	0.078755	0.009555	0.00	0.00
50 x 50	0.082351	0.007044	0.00	0.00
100 x 100	0.085978	0.005911	0.69	0.00
200 x 200	0.083166	0.005745	1.00	0.01
500 x 500	0.083219	0.006118	1.00	0.09
1000 x 1000	0.084688	0.005947	1.00	0.36
RPy 1D KS	0.081797	0.006348	-	0.46

Table 3 gives the results of comparing these two samples within ROOT at various binnings. All tests returned a probability of 0.0 that the samples came from the same distribution. It is easily seen that the limit of the two pseudo-CDFs at fine binnings is the CDF of each set of coordinates considered separately, as shown in Table 3 where the 1D KS test from the statistics package RPy[19] is applied to the individual sets of χ^2 and η data.

For completeness, Table 3 also gives the results of ROOT's 2D χ^2 -test between the histograms. Beyond moderate binning this test tends to indicate that the histograms are probably from the same distribution (probability=1.0). This is presumably due to the aforementioned "curse of dimensionality" – since most bins are empty, the histograms appear similar.

6. Parallel algorithms

The algorithms for Peacock's test discussed in Section 2 have trivial parallel versions. The brute-force algorithm can be parallelised by firstly having one parallel step where each one of p processors computes Peacock's statistic over $4n^2/p$ quadrants. The following trivial step consists of a parallel reduction operation that computes the maximum of the p statistics, directing it to one master processor. This gives an algorithm with time $O(\frac{n^2}{p} \lg n)$ and work $O(n^2 \lg n)$, making it an "optimal" parallel brute-force algorithm.

The parallelisation of the range-counting tree algorithm consists of three parallel steps. In the first step, p processors, in $O(n \lg n)$ time, build a copy of the range-counting tree over the n input points. In the second parallel step, each processor computes Peacock's statistics over $4n^2/p$ quadrants, using two-sided range-tree queries, in $O(\frac{n^2}{p} \lg n)$ time. The last step is again a reduction step which takes $O(\lg p)$ parallel steps to reduce the maximum to the master processor. The whole algorithm is asymptotically optimal, taking $O(\frac{n^2}{p} \lg n)$ time, but it performs more work than its sequential counterpart, the extra work being performed in the $p-1$ extra copies of the range tree generated in the first step.

Table 4 shows results and running times for the parallel implementation of both the brute-force and range-counting tree algorithms. For each test we used two samples of χ^2 versus η of the same size, one taken from aligned reconstruction data and the other from misaligned reconstruction data. All tests yielded identical results regardless of the number of processors, and the speedup was linear in the number of CPUs for both methods. As predicted in Section 2 the times for each method scale with sample size as n^3 and $n^2 \lg n$, respectively.

Table 4. Results and execution times for the parallel versions of Peacock's test. *Programmes in MPI C on a SUSE 9.3 64-bit Linux cluster with 1.8 GHz Opteron 265 worker nodes.*

Sample Size	Processors	Brute Force		Range-Counting Tree	
		Distance	Time (s)	Distance	Time (s)
4096	8	0.075195	2399	0.075195	127
4096	16	0.075195	1203	0.075195	63.2
4096	32	0.075195	601	-	-
4096	64	-	-	0.075195	16.0
5120	8	0.078320	4699	0.078320	214
5120	16	0.078320	2349	0.078320	106
5120	32	0.078320	1175	0.078320	53.5
6144	8	0.084147	8117	0.084147	327
6144	16	0.084147	4060	0.084147	166
6144	32	0.084147	2029	0.084147	82.0
6144	64	-	-	0.084147	41.1
7168	8	0.086356	12872	0.086356	463
7168	16	0.086356	6445	0.086356	235
7168	32	0.086356	3223	0.086356	117
7168	64	-	-	0.086356	58.8
8192	8	0.083374	19232	0.083374	650
8192	16	0.083374	9616	0.083374	322
8192	32	0.083374	4810	0.083374	162
8192	64	-	-	0.083374	82.2

7. Conclusion

These comparisons show that a proper comparison of 2D data can be difficult and expensive to achieve. For unbinned data the Fasano and Franceschini method has a clear advantage over Peacock's method, particularly when implemented with range-counting trees. Unfortunately, the performance boost afforded to Peacock's method by parallel implementations is not realisable by the Fasano and Franceschini algorithm, so there appears to be little scope to extend it beyond 10^5 points while retaining a reasonable turnaround. Other methods which may return speed-ups in parallel implementations are being investigated. Cooke's method, while undoubtedly fast, suffers from the shortcomings outlined earlier although it may still be useful in individual cases providing its limitations are recognised.

On the other hand, when comparisons need to be made on already-histogrammed data, there is currently no substitute for the ROOT 2D KS test. It especially wins out on speed when normal binning ranges are used but again its shortcomings need to be kept in mind. An alternative test within the root framework, using a "minimum energy" concept[20], is under development and should be available in the near future.

Acknowledgments

The authors would like to thank the CMS Tracker community and CMSSW developers for providing the tools used to produce the data sets. This work has been funded by the Science and Technology Facilities Council, UK.

References

- [1] Gagunashvili N 2006. χ^2 test for comparison of weighted and unweighted histograms. *Statistical Problems in Particle Physics, Astrophysics and Cosmology, Proc. of PHYSTAT05*, Oxford, UK, 12-15 September 2005 (London: Imperial College Press) pp 43-4
- Gagunashvili N 2006 Comparison of weighted and unweighted histograms [arXiv:physics/0605123](https://arxiv.org/abs/physics/0605123)
- [2] Chakravati I M , Laha R G and Roy J 1967 *Handbook of Methods of Applied Statistics, Volume I* (New York: John Wiley and Sons) pp 392-4
- [3] Scott D W 1992 *Multivariate Density Estimation: Theory, Practice, and Visualisation* (New York: John Wiley & Sons Inc.)
- [4] CMS Collaboration 2006 *CMS Physics Technical Design Report: Volume I, Detector Performance and Software* ed D Acosta http://cmsdoc.cern.ch/cms/cpt/tdr/ptdr1_final_colour.pdf
- [5] Barbone L , De Filippis N, Buchmueller O, Schilling F-P, Speer T and Vanlaer P 2006 *Nucl. Instr. and Meth. A* **566** 45-9
- [6] Lopes R H C, Reid I and Hobson P R 2007 The two-dimensional Kolmogorov-Smirnov test. *Proc. XI Int. Workshop on Advanced Computing and Analysis Techniques in Physics Research* April 23-27 2007, Amsterdam (in press) or <http://bura.brunel.ac.uk/bitstream/2438/1166/1/acat2007.pdf>
- [7] Peacock JA 1983 Two-dimensional goodness-of-fit testing in astronomy *Mon. Not. R. Astron. Soc.* **202** 615-27
- [8] Lueker G S 1978 A data structure for orthogonal range queries *Proc. 19th IEEE Symp. on Foundations of Computer Science* 28-34
- [9] Arge L, Brodal G S, Fragerberg R and Laustsen M 2005 Cache-oblivious planar orthogonal range searching and counting *Proc. of the 21st Ann. Symp. on Computational Geometry* 160-9
- [10] Aho A V, Holcroft J E and Ullman J D 1974 *The Design and Analysis of Computer Algorithms* (Reading, MA: Addison-Wesley)
- [11] Fasano G and Franceschini A 1987 A multidimensional version of the Kolmogorov-Smirnov test *Mon. Not. R. Astron. Soc.* **225** 155-70
- [12] Press W H, Teukolsky S A, Vetterling W T and Flannery B P 2002 *Numerical Recipes in C++: The Art of Scientific Computing* (Cambridge: Cambridge University Press)
- [13] Knuth D E 1998 *The Art of Computer Programming, Volume 3: Sorting and Searching* (Reading, MA: Addison-Wesley)
- [14] Cooke A 1999 <http://www.acooke.org/jara/muac/algorithm.html>
- [15] Chan I 2002 Parallelizing a 2D Kolmogorov-Smirnov statistic
<http://beowulf.lcs.mit.edu/18.337-2002/projects-2002/ianchan/KS2D/Project%20Page.htm>

- [16] Knuth D E 1998 *The Art of Computer Programming, Volume 1: Fundamental Algorithms* (Reading, MA: Addison-Wesley)
- [17] Cormen T H, Leiserson C E, Rivest R L and Stein C 2001 *Introduction to Algorithms* (Cambridge, MA: MIT Press)
- [18] <http://root.cern.ch>
- [19] <http://rpy.sourceforge.net>
- [20] Zech G and Aslan B 2003 A new test for the multivariate two-sample problem based on the concept of minimum energy. [arXiv:math/0309164](https://arxiv.org/abs/math/0309164)