MAUSAM, **66**, 3 (July 2015), 457-466

551. 509.5

Fast calculation of the fractions skill score

NATHAN FAGGIAN, BELINDA ROUX, PETER STEINLE and BETH EBERT

Research and Development, Australian Bureau of Meteorology, GPO Box 1289, VIC 3001, Melbourne, Australia

e mail : n.faggian@bom.gov.au

सार – पूर्वानुमान सत्यापन पद्धति को फ्रैक्शन स्किल स्कोर्स (एफ एस एस) के नाम से जाना जाता है जो स्लाइडिंग विंडो ऑपरेटर्स के उपयोग से विशेष प्रकार की गणना करता है जो खर्चीली गणना हो सकती है। इस स्कोर का मुख्य अवयव खंडित घटना आवृतियों की गणना करना है जो भारी-भरकम सब-ग्रिड्स (विंडोज़) योगफल के समतुल्य है। इसे आमतौर पर कुण्डलीकरण ऑपरेशन से जाना जाता है। इसकी वैकल्पिक व्यवस्था है कि 'योगफल क्षेत्र सारणी' का उपयोग किया जाए, जिसका इस्तेमाल कंप्यूटर ग्राफिक्स में क्षेत्र संरचना में सब-ग्रिड्स का तेजी से योगफल निकालने में किया जाता है। इस शोध पत्र में हमने एफ एस एस की गणना अवधि को काफी कम करने के लिए 'योगफल क्षेत्र सारणी' के उपयोग के बारे में वर्णन किया है, हमने इस स्कोर को सामान्य करने के लिए समय की अवधि को भी शामिल किया है। हमने आदर्श स्थिति में स्थानिक सत्यापन विधियों अन्तर-तुलनात्मक परियोजना से इस प्रणाली को दर्शाया है और उच्च विभेदन वाली एन डब्ल्यू पी डेटासेट पर स्कोर की विशेषताओं की व्याख्या की है।

ABSTRACT. The forecast verification metric known as the Fractions Skill Score (FSS) is typicallycomputed using sliding window operators, which can be computationally expensive. A keycomponent of the score is the computation of fractional event frequencies, which is equivalent to a weighted summation of sub-grids (windows) commonly realized as a convolutionoperation. An alternative approach is to use "summed area tables", which have been used incomputer graphics as a means to quickly compute summations of sub-grids in texture fields.In this paper we describe how a summed area table can effectively reduce the computationtime of the FSS while also allowing the score to generalize to include the time dimension.We demonstrate the methodology on idealized cases from the Spatial Verification MethodsInter-comparison Project and explore the properties of the score on a high-resolution NWPdataset.

Key words– NWP, Fractions skill score (FSS).

1. Introduction

As numerical weather prediction (NWP) model forecasts increase in spatial and temporal resolution, the community is turning to diagnostic spatial verification approaches to provide moreuseful and meaningful quantitative evaluation than is possible using simple statistics like rootmean square error. Gilleland *et al.* (2009) identified four types of spatial verification methods, namely neighborhood methods, scale separation methods, feature-based methods, and field de-formation methods, each of which is well suited for particular types of verification problems. The neighborhood methods are particularly useful for determining the spatial scales at which sufficient forecast skill is achieved. They use various metrics to compare forecasts to observations in spatial windows of progressively larger size, thereby providing information aboutforecast accuracy as a function of spatial scale.

The fractions skill score (FSS) introduced by Roberts and Lean (2008) is a neighborhood verification method being used by the U.K. Met. Office and other national centres to verify high resolution precipitation forecasts against radar rainfall estimates (Mittermaier and Roberts, 2010). Conceptually, the spatial distribution of events within a small area is treated probabilistically rather than deterministically. An event is the binary (yes or no) occurrence of something, for example, whether rain in a grid box exceeds a certain intensity threshold. For a given $n \times n$ window size, the FSS considers a perfect forecast to be one with the same frequency of eventsas was observed within the window, regardless of their particular placement within the window. An example of an observation and forecast field that could yield a perfect FSS is shown in Figs. 1(a&b). The fractions skill score is computed as the fractions Brier score (a variation of the Brier skill score used to verify probability forecasts), divided by the sum of the mean squared forecast and observed fractions and can be written as:

$$FSS = 1 - \frac{\frac{1}{N}\sum_{i=1}^{N}\left(p_f - p_o\right)^2}{\frac{1}{N}\sum_{i=1}^{N} p_f^2 + \frac{1}{N}\sum_{i=1}^{N} p_o^2} \qquad (1)$$

where N is the number of windows in the domain (dependent on sliding window size) and p_f is the forecast fraction, p_o is the observed fraction of the sliding window.

(a) Observation

1	0	0	0
0	0	1	1
0	1	0	0
0	0	1	0
1	0	1	0

(b Forecast

0	0	1	1
0	0	0	0
1	0	0	0
1	0	0	1
1	0	0	1

Figs. 1(a&b). Two fields (observation and forecast) that have an equivalent fractional event frequency and lead to a perfect FSS (assuming a single window)

(a) Field, O

1	0	0	0
0	1	1	0
0	1	1	0
0	1	0	0
0	0	0	0

(b Summed Area Table, $\hat{O}$

1	1	1	1
1	2	3	3
1	3	5	5
1	4	6	6
1	4	6	6

Figs. 2(a&b). An example of a field (a) and the corresponding summed area table (b)

The computational cost of the FSS is linked to the method used to compute the fractional event frequency (p_o, p_f), the field domain size and the window size. Typically the FSS is computed for a large range of window sizes and the resulting score plotted as a function of windowsize. This allows the scale at which the FSS reaches the target skill of $\left(0.5+\frac{p_o}{2}\right)$ to be easily determined [Roberts and Lean (2008)].

2. Fast methods

As defined by Roberts and Lean (2008) the fractional event frequency of a window with width *n*, centered on the coordinate (*i*,*j*) in the observation field *O* is computed as:

$$O(n)(i,j)=\frac{1}{n^2}\sum_{k=1}^{n}\sum_{l=1}^{n}I_o\left[i+k-1-\frac{(n-1)}{2},\ j+l-1-\frac{(n-1)}{2}\right] \quad (2)$$

where $I_o(i,j)$ is either 0 or 1. The equation is also applied to the forecast field *F*, with asimple matrix substitution (I_o for I_F). In both cases the equation is evaluated over all pixelsin the field, requiring (n × n) additions in each centered window; as the window size increasesthe computational load also increases. In practice this can be implemented using a convolutionoperation, for example using a box-car kernel.

2.1. *Summed area table (integral image)*

In the context of the FSS tables, the event frequency calculation is repeated for multiple windowsizes and object threshold levels, easily making it the most time consuming component of computing the spatial verification metric. Even though convolution is often optimized on moderncomputers, calculating the score can be slow. In this paper we propose the use of a "summedarea table", also known as an integral image, which is a structured array that is pre-computed, then indexed to efficiently compute summations over a field.

The concept of a summed area table was first introduced for the efficient calculation of "mip-maps" in a computer graphics (Crow, 1984) and they are still popular in computer vision for fast feature extraction in object detection frameworks (Viola and Jones, 2004). To computethe summed area table, $\hat{O}$, the value at any point (*i*,*j*) is the sum of all the grid cells aboveand to the left of (*i*,*j*), inclusive:

$$\hat{O}(i,j)=\sum_{\hat{i}=0}^{\hat{i}<i}\sum_{\hat{j}=0}^{\hat{j}<j}O(\hat{i},\hat{j}) \quad (3)$$

Once the summed area table has been computed, the task of summing a window of width mand height *n* from the coordinate (*i*,*j*) in the observation field *O* is accomplished with just fourarray references using the following relation:

$$\sum_{\hat{i}=i-m}^{i=m}\sum_{\hat{j}=j-n}^{j+n}O(\hat{i},\hat{j})=\hat{O}(i,j)+\hat{O}(i+m,j+n)-\hat{O}(i+m,j)-\hat{O}(i,j+n) \quad (4)$$

The calculation of $\hat{O}$ is demonstrated in Figs. 2(a&b) where the right panel shows the summedarea table with the coordinates for each grid-cell. Using the same relation we can define Equation 2, extending the fractional event frequency calculation to rectangular windows (m × n) centeredon the coordinates (*i*,*j*) like so:

$$O(m,n)(i,j)=\frac{1}{m\times n}\left[\hat{O}\left(i-\frac{m}{2},j-\frac{n}{2}\right)+\hat{O}\left(i+\frac{m}{2},j+\frac{n}{2}\right)-\hat{O}\left(i+\frac{m}{2},j\right)-\hat{O}\left(i,j+\frac{n}{2}\right)\right] \quad (5)$$

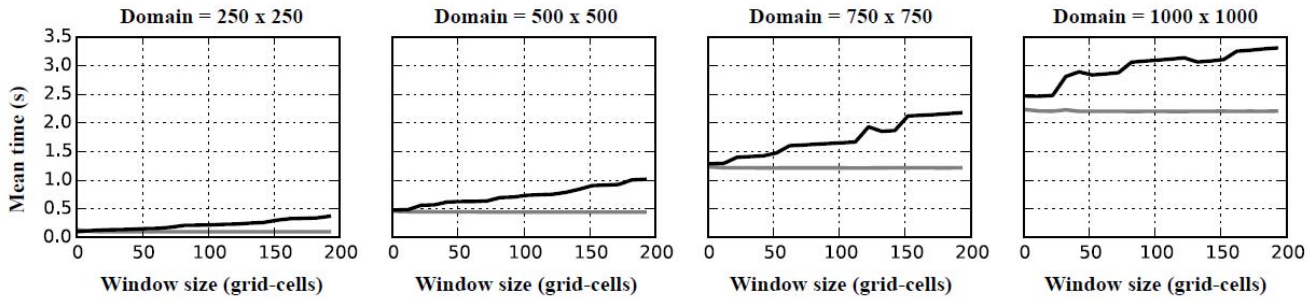


Fig. 3. The mean computation time (100 repetitions) for the FSS when using a Fourier Transform based convolution (black) and summed area tables (gray) is shown for increasing square domains (250 to 1000 grid-cells) and window sizes (2 to 200 grid-cells)

where $\hat{O}$ is the precomputed summed area table of the observation field, using Equation 3. Asbefore the equation is also applied to the forecast field F, with a simple field substitution (F for O), requiring two summed area tables to be computed for each calculation of the fractionskill score. Although the FSS is normally computed for square windows, we generalize inEquation 4 to rectangular windows to facilitate extension to the time domain (see Appendix) or non-isotropic domains such as coasts or mountain ranges, where unequal dimensions may bepreferable.

2.1.1. *Sampling summed area tables*

In practice the summed area table is computed once per thresholdded field, as are the shiftedcoordinates of the entire domain and indexing the table with a matrix of coordinates performs (as shown above) performs a fast sliding window sum. Using a matrix notation for sampling wedefine the matrix sampling function as {} and let the entire set of domain coordinates i and j be represented as the matrices I and J respectively:

$$\hat{O}(I,J)=\begin{bmatrix}\hat{O}(i_0,j_0) & \dots & \hat{O}(i_0,j_m)\\ \vdots & \ddots & \vdots\\ \hat{O}(i_n,j_0) & \dots & \hat{O}(i_n,j_m)\end{bmatrix} \tag{6}$$

which we can use to compute a sliding window sum using shifted coordinates samples, like so:

$$O(m,n)(I,J)=\frac{1}{m\times n}\left[\hat{O}\left(I-\frac{m}{2},J-\frac{n}{2}\right)+\hat{O}\left(I+\frac{m}{2},J+\frac{n}{2}\right)-\hat{O}\left(I+\frac{m}{2},J\right)-\hat{O}\left(I,J+\frac{n}{2}\right)\right] \tag{7}$$

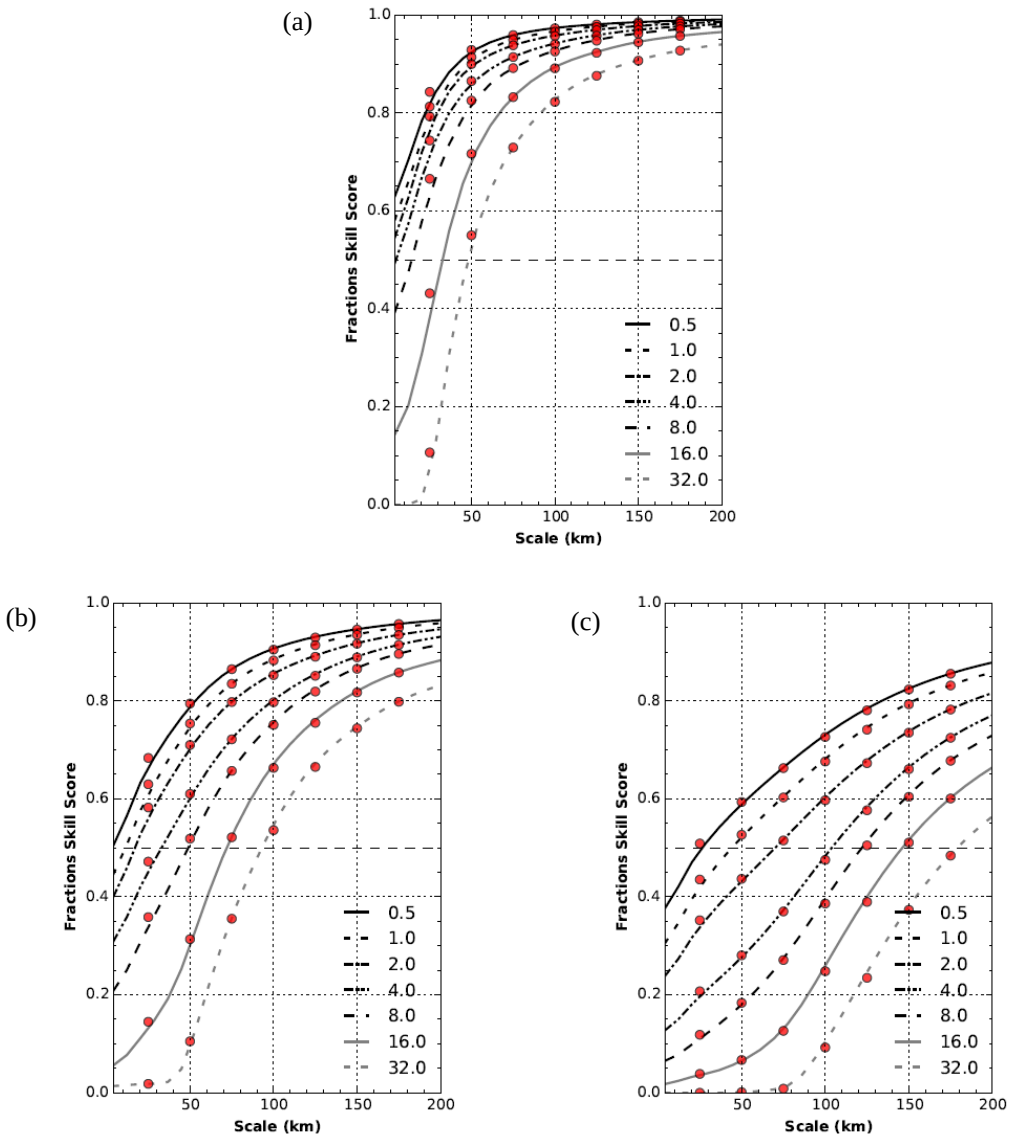
The computational cost of sampling different window sizes is always the same because thesummed area table is computed only once and the number of coordinate samples does notincrease. The calculation is only bound by the gridded forecast domain size, which increasesthe size of the coordinate matrices.

2.2. *Boundary conditions*

As windows approach domain boundaries there will be boundary effects and window scalesmust be constrained to a fraction of the maximum extent of the domain to minimize them. Window coordinates that extend beyond the indexable regions of the summed area table mustbe mapped to values. Our approach clips the coordinates of the rectangles that extend beyondthe forecast domain, which is equivalent to padding the field with zeros (as done by Roberts and Lean (2008).

2.3. *Complexity and computation time*

The summed area table method for calculating the sliding window sum required for the FSS isvery efficient, even when compared to optimized FFT based convolution. Fig. 3 demonstratesthe difference between the two approaches, showing that small window sizes both methods arefast to compute. However, the integral image based FSS exhibits (approximate) constant time performance and as window sizes increase the maximum performance gain ranges between 30% to 85% relative to the FFT approach. A fair and objective way to compare algorithm complexityis through the use of Big “O” notation (Knuth, 1968). The notation is a measure of algorithm properties relative to their growth rates: non-linear growth is considered poor while linear or approximately linear is considered good. Using the integral image to calculate requires aconstant number of array lookups. The calculation is bound by the number of grid cells (N) and a constant number (4) of lookups for a variable sized window, resulting in a computational complexity of $O(N)$. The more complex convolution approach is a combination of both a Fast Fourier

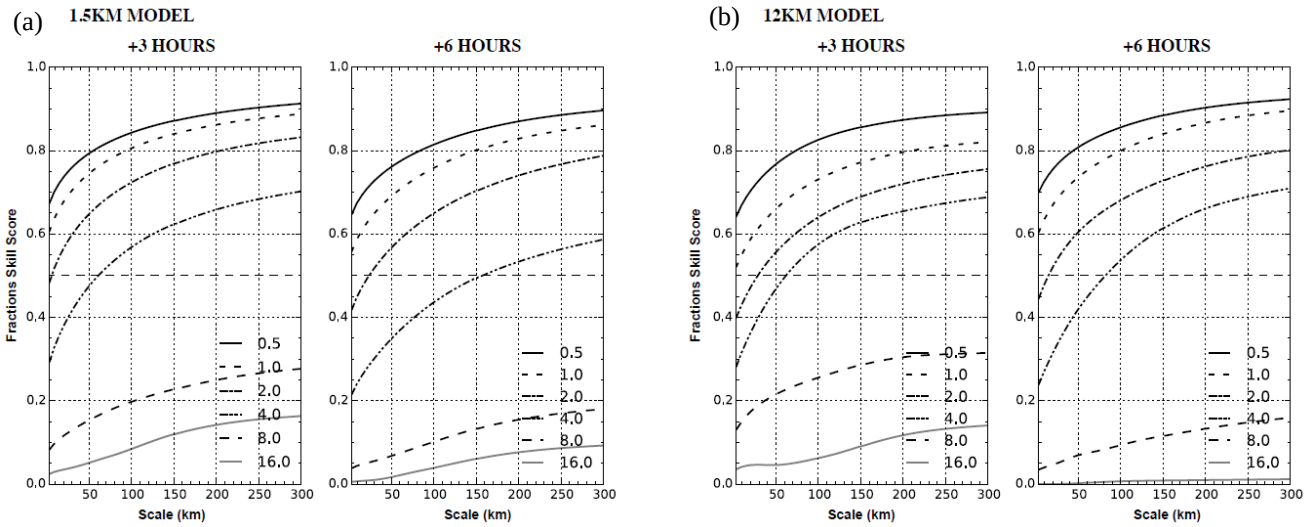


Figs. 4(a-c). Fast FSS calculations shown at precipitation thresholds (0.5 to 32 mm/hr) with markings to show (red circles) a corresponding subset of the calculations from the NCAR implementation (Gilleland, 2014). The examples (a, b, c) are from the perturbed precipitation case set provided by the Spatial Verification Inter-comparison Project

Transformation (FFT) $O(N \log N)$ and a matrix multiplication $O(N_2)$, which explainsthe differences seen in Fig. 3.

2.4. *Validation of the approach*

To validate the method it was applied to a precipitation data set provided by the Spatial Verification Inter-comparison Project (Ahijevych*et al*., 2009). Figs. 4(a-c) shows the Fast FSS computed for a subset of the perturbed precipitation cases (as lines) which are compared tovalues (red circles) from the NCAR FSS implementation developed by Gillelan*det al*. (2014) atprecipitation thresholds of 0.5 to and 32 mm/hr and scale intervals of 25 km. Figs. 4(a-c) clearly shows the equivalence of the fast FSS calculation to the original methodology proposed by Roberts and Lean (2008).



Figs. 5(a&b). Fractions Skill scores for (a) experimental 1.5 km high-resolution and (b) 12 km operational NWP systems at different spatial scales and precipitation threshold levels

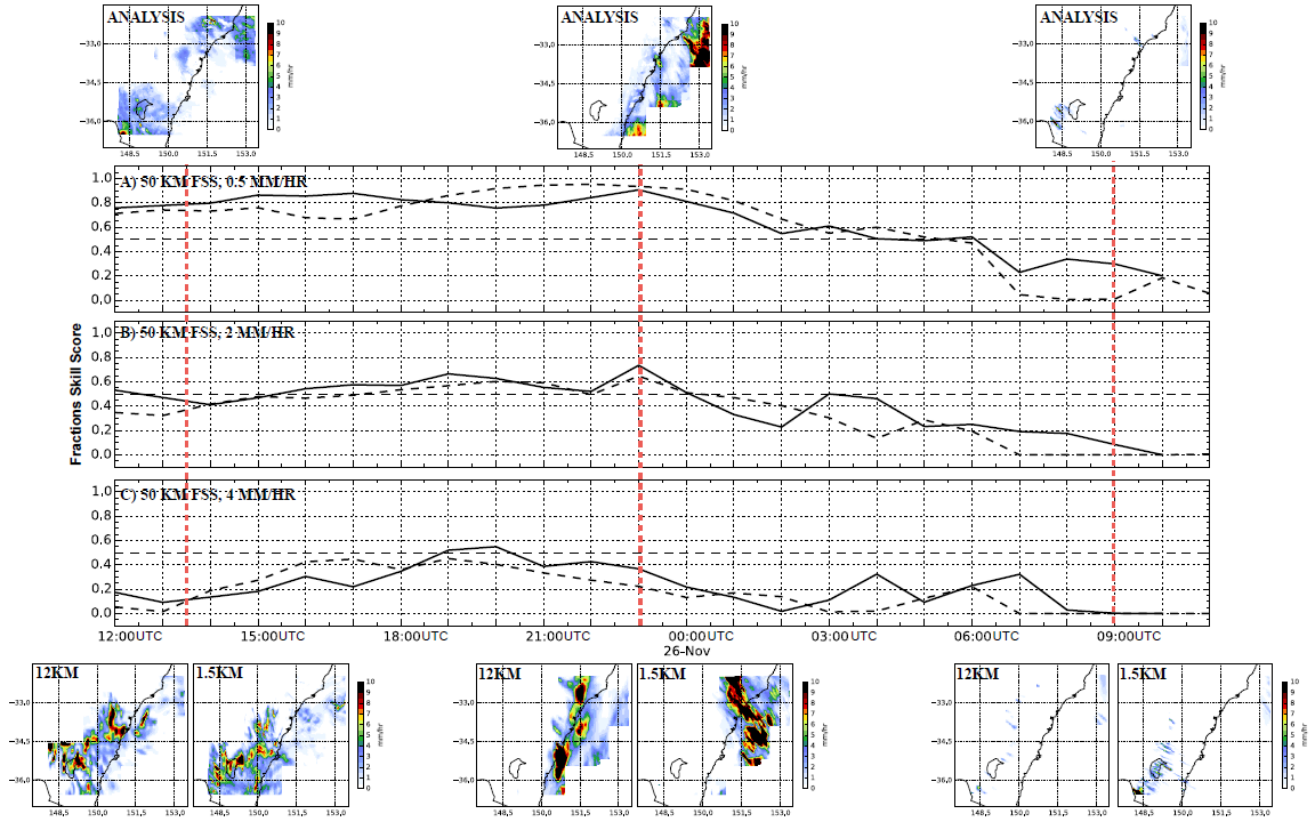


Fig. 6. FSS for 1.5 (solid) and 12 (dashed) km resolution NWP, for three precipitation thresholds. Subsets of the NWP models and corresponding analysis fields show the spatial variation at the beginning, middle and end of the period centered on 0000 UTC 26th of November, 2011

3. Demonstration using Australian rain events

As a further demonstration of the method, we compare the performance of an operational 12 km regional NWP system that includes 6 hourly four dimensional variational assimilation and an experimental 1.5 km system that includes 3 hourly three dimensional variation assimilation. The 12 km system uses routine *in situ* and satellite data, as described in the Bureau of Meteorology's Operations Bulletin 98 (BoM, 2013). The 1.5 km system is based on the U. K. Met. Office 1.5 km system (Lean *et al*., 2008) and applied to a smaller 10° × 12° domain centered over Sydney in southeastern Australia. The higher resolution system also has higher vertical resolution near the Earth's surface, at the expense of levels in the mesosphere. For this reason the high resolution system assimilates a reduced set of channels from satellite sounders. The high resolution system is however nudged towards rainfall analyses based on radar and rain gauge data (Chumchean *et al*., 2006) during the assimilation window. The forecasts are verified against analyses based on radar rainfall estimates and rain gauge data (Chumchean *et al*., 2006) where the radar coverageis of a suitable quality.

3.1. *Comparing multiple rain-events*

From the period of November 2011 to June 2012 hourly forecasts of precipitation were compared between the 1.5 km and 12 km models. Events of interest were identified from the dataset by selecting forecasts that maintained overland precipitation greater than 1 mm/hr, resulting in a large set (many thousands) of hourly forecasts for analysis. The FSS results for forecasts of hourly rainfall out to 36 hours were then grouped by lead time and the mean fraction skill score was calculated by averaging the numerator and denominator of the score separately.

Figs. 5(a&b) shows the resulting skill plots for spatial scales up to 300 km and precipitation thresholds from 0.5 mm/hr to 16 mm/hr at lead times of three and six hours respectively. The results suggest that the 1.5 km model performs slightly better at low precipitation levels andspatial scales, particularly at the three hour lead time which is likely due, at least in part, tothe assimilation of radar data. As the forecast lead times increase the skill of the 12 km model appears to improve while the skill of the 1.5 km model decreases. At the six hour lead timethe 1.5 km model appears to reduce in skill more rapidly than the 12 km model at moderateprecipitation thresholds.

3.2. *Comparing a single event over time*

We also performed a temporal analysis of an intense rain event during the 26th of November, 2011 for 1.5 km and 12 km model forecasts initialized at 1200 UTC on the 25th. For each forecastwe calculated the FSS for the 50 km spatial scale and 0.5, 2.0 and 4.0 mm/hr precipitation thresholds. We selected these scales to explore the differences between the models at low rainrates and to assess the temporal skill of the forecasts for this event. Results are shown in Fig. 6.

Over the period of interest it is difficult to identify where one model performed significantly better than the other. The high values of FSS in Fig. 6 indicate that both models accurately captured the occurrence of rain. The peak skill occurred near the middle of the event. Interestingly, both models skill slowly decrease at approximately the same rate toward the endof the period, which may result from both a reduction in the rain rate over the period andthe expected reduction of forecast skill as lead-time increases. The skill of both forecastswas least for the heaviest rain thresholds (as must be the case for FSS), with the 1.5 km model showing some advantage over the coarser model for most of the period.

4. Conclusions

A key component of the FSS is the computation of fractional event frequencies, which is equivalent to a weighted summation of sub-grids (windows). In this paper we have demonstrated how summed area tables can be used as an efficient way to calculate the FSS and validated themethod against independent calculations of FSS on a high-resolution data set. The key benefitof using summed area tables over other approaches is their comparative simplicity and, most importantly, their speed. Using the FSS we compared two NWP models at 36 matched lead times, leading to the calculation of many thousands of the FSS, to explore mean and temporal performance of the models. Without a fast approach to calculate the FSS, as we have provided, generation of the scores required for our verification experiments would take a significantly longer time or a large computer resource. As suggested earlier the FSS can be extended to the time dimension (Duc *et al*., 2013) and the fast computation of the FSS for three dimensional data *via* summed area volumes is described in the Appendix. This work has shown that summed area tables are a useful tool when calculating verification metrics that require local summation (or smoothing), for interested readers a good reference implementation for summed area tablesis provided by van der Walt *et al*. (2014).

References

Ahijevych, D., Gilleland, E., Brown, B. G. and Ebert, E. E.,2009, "Application of Spatial Verification methods to idealized and NWP-Gridded precipitation forecasts", *Weather and Forecasting*, **24**, 6, 1485-1497.

BoM, 2013, "APS1 upgrade of the ACCESS-R Numerical Weather Prediction system", Technical Report 98, Bureau of Meteorology National Meteorological and Oceanographic Centre,URL http://www.bom.gov.au/australia/charts/bulletins/apob98.pdf (last access: July, 2014).

Chumchean, S., Sharma, A. and Seed, A., 2006, "An integrated approach to error correction for realtime radar-rainfall estimation", *Journal of Atmospheric and Oceanic Technology*, **23**, 1, 67-79.

Crow, F. C., 1984, "Summed-area tables for texture mapping",*ACM SIGGRAPH Computer Graphics*,**18**, 3, 207-212.

Duc, L., Saito, K. and Seko, H., 2013, "Spatial-temporal fractions verification for high-resolution ensemble forecasts", *Tellus A*, **65**.

Gilleland, E., 2014, "Spatial Vx: Spatial Forecast Verification, 2014", URL http://www.ral.ucar.edu/projects/icp. R package version 0.2-1.

Gilleland, E., Ahijevych, D, Brown, B. G.,Casati, B. and Ebert, E. E., 2009, "Intercomparison of spatial forecast verification methods",*Weather and Forecasting*, **24**, 5, 1416-1430.

Knuth, D., 1968, "The Art of Computer Programming 1: Fundamental Algorithms 2: Semi numerical Algorithms 3: Sorting and Searching", MA : Addison-Wesley.

Lean, H. W., Clark, P. A., Dixon, M., Roberts, N. M., Fitch, A., Forbes, R. and Halliwell, C., 2008,"Characteristics of high-resolution versions of the met office united model for forecasting convection over the united kingdom", *Mon.Wea. Rev.*, **136**, 9, 3408-3424.

Mittermaier, M. and Roberts, N., 2010, "Intercomparison of spatial forecast verification Methods : Identifying skillful spatial scales using the fractions skill score", *Weather and Forecasting*, **25**, 1, 343-354.

Roberts, N. M. and Lean, H. W., 2008, "Scale-Selective verification of rainfall accumulations from high-resolution forecasts of convective events", *Mon. Wea. Rev.*, **136**, 1, 78-97.

Tapia, E., 2011, "A note on the computation of high-dimensional integral images", *Pattern Recognition Letters*, **32**, 2, 197-201.

Van der Walt, S., Schonberger, J. L., Nunez-Iglesias, J. Boulogne, F., Warner, J. D., Yager, N., Gouillart, E. and Yu, T., 2014, "Scikit-image : Image processing in python", *Technical report*, Peer J. Pre Prints.

Viola, P. and Jones, Robust, M. J., 2004, "Real-time face detection", *International Journal of Computer Vision*, **57**, 2, 137-154.

APPENDIX

Summed area volume (integral volume)

A benefit of summed area tables is that the same approach also applies to forecast volumes.For demonstration only, we follow the definition by Tapia (2011) to extend the FSS calculationto include the third (time) dimension, using the "integral volume". To compute the integral volume ($\hat{O}$) the value at any point (i,j,k) is computed as:

$$\hat{O}(i,j,k) = \sum_{\hat{i}=0}^{\hat{i}<i} \sum_{\hat{j}=0}^{\hat{j}<j} \sum_{\hat{k}=0}^{\hat{k}<k} O\left(\hat{i},\hat{j},\hat{k}\right) \tag{8}$$

As before the task of summing a volume of dimension $m \times n \times p$, centered on the coordinate (i,j,k) is accomplished in constant time with just eight array references, leading to a redefinitionof Equation 2:

$$O(m,n,p)(I,J,K) = \frac{1}{m \times n \times p} \begin{bmatrix} \hat{O}(I+m, J+n, K+p) - \\ \hat{O}(I-m, J+n, K+p) - \\ \hat{O}(I+m, J-n, K+p) - \\ \hat{O}(I+m, J+n, K-p) - \\ \hat{O}(I-m, J-n, K+p) + \\ \hat{O}(I-m, J+n, K-p) + \\ \hat{O}(I+m, J-n, K-p) - \\ \hat{O}(I-m, J-n, K-p) \end{bmatrix} \tag{9}$$

where the coordinates are (*I*,*J*, *K*) are defined as matrices and the sampling of the integralvolume is performed in a similar fashion to Equation 7. In practice the time window (*p*) for meteorological forecasts will generally be small since the time scales relevance for mesoscale modeling are typically on in the order of hours. For the particular case of estimating rainfall accuracy, combining time and spatial uncertainty has the disadvantage of making interpretation of the results more difficult. However, if the forecast user is willing to be tolerant of some temporal error in forecast mesoscale weather processes, then the addition of a temporal dimensionin the skill score is appropriate.

FSS implementation

An example reference implementation is provided to demonstrate the simplicity of using thesummed area table calculations as a part of computing the FSS. For comparison a FFT basedapproach is also provided.

Implementation 1: FSS using summed area tables and FFT based convolution in Python.

```
"""
.. module:: fss
:platform: Unix
:synopsis: Compute the fraction skill score (2D).
.. moduleauthor:: Nathan Faggian <n.faggian@bom.gov.au>
"""

import numpy as np
import pandas as pd
from scipyimport signal

defcompute_integral_table(field):
    return field.cumsum(1).cumsum(0)

deffourier_filter(field, n):
    return signal.fftconvolve(field, np.ones((n, n)))

defintegral_filter(field, n, table=None):
    """
    Fast summed area table version of the sliding accumulator.
    :param field: nd-array of binary hits/misses.
    :param n: window size.
    """
    w = n // 2
    if w < 1.:
       return field
    if table is None:
       table = compute_integral_table(field)

    r, c = np.mgrid[0:field.shape[0], 0:field.shape[1]]
```

```
    r = r.astype(np.int)
    c = c.astype(np.int)
    w = np.int(w)

    r0, c0 = (np.clip(r - w, 0, field.shape[0] - 1),
       np.clip(c - w, 0, field.shape[1] - 1))
    r1, c1 = (np.clip(r + w, 0, field.shape[0] - 1),
       np.clip(c + w, 0, field.shape[1] - 1))

    integral_table = np.zeros(field.shape).astype(np.int64)
    integral_table += np.take(table, np.ravel_multi_index((r1, c1), field.shape))
    integral_table += np.take(table, np.ravel_multi_index((r0, c0), field.shape))
    integral_table -= np.take(table, np.ravel_multi_index((r0, c1), field.shape))
    integral_table -= np.take(table, np.ravel_multi_index((r1, c0), field.shape))
    returnintegral_table


deffourier_fss(fcst, obs, threshold, window):
    """
    Compute the fraction skill score using convolution.
    :paramfcst: nd-array, forecast field.
    :paramobs: nd-array, observation field.
    :param window: integer, window size.
    :return: tuple of FSS numerator, denominator and score.
    """
    fhat = fourier_filter(fcst> threshold, window)
    ohat = fourier_filter(obs> threshold, window)
    num = np.nanmean(np.power(fhat - ohat, 2))
    denom = np.nanmean(np.power(fhat, 2) + np.power(ohat, 2))
    return num, denom, 1.-num/denom

deffss(fcst, obs, threshold, window, fcst_cache=None, obs_cache=None):
    """
    Compute the fraction skill score using summed area tables.
    :paramfcst: nd-array, forecast field.
    :paramobs: nd-array, observation field.
    :param window: integer, window size.
    :return: tuple of FSS numerator, denominator and score.
    """
    fhat = integral_filter(fcst> threshold, window, fcst_cache)
    ohat = integral_filter(obs> threshold, window, obs_cache)
```

```
    num = np.nanmean(np.power(fhat - ohat, 2))
    denom = np.nanmean(np.power(fhat, 2) + np.power(ohat, 2))
    return num, denom, 1.-num/denom

deffss_frame(fcst, obs, windows, levels):
    """
    Compute the fraction skill score data-frame.
    :paramfcst: nd-array, forecast field.
    :paramobs: nd-array, observation field.
    :param window: list, window sizes.
    :param levels: list, threshold levels.
    :return: list, dataframes of the FSS: numerator,denominator and score.
    """
    num_data, den_data, fss_data = [], [], []

    for level in levels:
       ftable = compute_integral_table(fcst> level)
       otable = compute_integral_table(obs> level)

       _data = [fss(fcst, obs, level, w, ftable, otable) for w in windows]

       num_data.append([x[0] for x in _data])
       den_data.append([x[1] for x in _data])
       fss_data.append([x[2] for x in _data])

    return (pd.DataFrame(num_data, index=levels, columns=windows),
            pd.DataFrame(den_data, index=levels, columns=windows),
            pd.DataFrame(fss_data, index=levels, columns=windows))
..
```