Application of sequential testing problem to online detection of transient stability status for power systems

Jhonny Gonzalez*, Yerkin Kitapbayev†, Tingyan Guo**,
Jovica V. Milanović**, Goran Peskir*, & John Moriarty‡

***Abstract*— We address the problem of predicting the transient stability status of a power system as quickly as possible in real time subject to probabilistic risk constraints. The goal is to minimise the average time taken after a fault to make the prediction, and the method is based on ideas from statistical sequential analysis. The proposed approach combines probabilistic neural networks with dynamic programming. Simulation results show an approximately three-fold increase in prediction speed when compared to the use of pre-committed (fixed) prediction times.**

I. INTRODUCTION

The collection of real-time and near real-time data from phasor measurement units (PMUs) has enabled the development of online decision support algorithms for power system applications. An important example is the real-time prediction of post-fault transient stability status following a power system contingency. In this problem of *stability classification*, decision rules may be trained off-line using representative samples of stable and unstable contingencies and then used to predict the stability class of new contingencies in real time. Several methodologies have been applied in the literature, including decision trees [1], [2], [3], [4], [5], [6], support vector machines [7], [8], [9], and artificial neural networks [10], [11], [12], [13]. While the binary classification problem (where the goal is simply to distinguish between stable and unstable contingencies) has been more widely studied, more detailed techniques have also been developed to distinguish between different classes of instability [2], [13], [14].

In the immediate aftermath of a particular fault, the accuracy of stability predictions should increase over time. On the other hand, rapid post-fault action is necessary to enable the application of appropriate corrective control. Thus there is a balance to be struck in determining *when to act* on such online predictions. This balance does not seem to have been addressed in the literature, in particular considering the availability of real-time wide area measurement data from PMUs. Accordingly, in the present paper we address this gap by formulating the stability classification problem as one of sequential testing (see, for example, [15]) under risk constraints.

*School of Mathematics, The University of Manchester, Manchester, M13 9PL, UK. †Questrom School of Business, Boston University, Boston, MA 02215, US. **School of Electrical and Electronic Engineering, The University of Manchester, Manchester, M60 1QD, UK ‡School of Mathematical Sciences, Queen Mary University of London, London, E1 4NS, UK j.moriarty@qmul.ac.uk

Our aim is twofold: firstly to develop an online *probabilistic* prediction of the stability class, and secondly to derive a decision rule which acts on these predictions *at the optimal time*. The risk constraints express the required level of accuracy by placing limits on the rate of stability prediction errors of various kinds.

The optimal time we seek is not fixed, but may vary depending on the post-fault PMU measurements. Thus although our decision rule is computed offline, the time of action is determined flexibly and appropriately in real time rather than being pre-committed. In this way clear initial information can be acted upon quickly and, conversely, action can be delayed when the initial post-fault data is ambiguous. This approach is enabled by the combination of probabilistic neural networks (PNNs, see for example [16], [17]) to provide an online indication of how informative are the post-fault measurements, together with dynamic programming which enables the optimal decision rules to be computed efficiently. In particular, by *waiting* until sufficient information is available, this approach aims to achieve the fastest possible predictions on average while maintaining the same accuracy as significantly longer pre-committed action times.

II. PROBLEM FORMULATION

Consider a power system with generators labelled $i = 1, \ldots, g$. Suppose that a transient disturbance has been cleared at time t_1, and that the rotor angle for each generator is observed until the later time $t_M > t_1$. Let $\delta_i(t)$ denote the rotor angle of generator i at time t, and $\delta(t) = (\delta_1(t), \ldots, \delta_g(t))$ denote the vector of all rotor angles at time t. Our objective is to use as few of these observation vectors as possible to make an acceptable *prediction* of the system's post-disturbance response over the entire time interval $[t_1, t_M]$. More precisely the goal is to make this prediction as quickly as possible on average, so that appropriate corrective control may be applied at the earliest opportunity, while respecting limits on the frequency of incorrect predictions.

The decision problem is the following. We suppose that contingencies can be classified as either stable or unstable, and that each unstable contingency can be further classified within a finite set of different unstable classes. (In [1] and [2], for example, a contingency is said to be unstable if the absolute difference between the rotor angles of any two generators exceeds 360° within a certain time horizon.) Further we suppose that the unstable contingencies can be classified into I different types. For convenience we represent

the set of stability classes as $\mathcal{C} = \{0, 1, \ldots, I\}$, where 0 is the stable class and the positive integers are labels for the I unstable classes.

Our problem may be represented as the choice of a pair of decision variables (τ, d), where $t_1 \leq \tau \leq t_M$ is the time after the disturbance at which we 'stop' observing the process $\delta = (\delta(t))_{t_1 \leq t \leq t_M}$ and make the prediction d regarding the transient stability status. Both the *stopping time* τ and the prediction d must be determined using *only* the process $(\delta(t))_{t_1 \leq t \leq \tau}$, i.e. observations of the process δ up to the stopping time τ itself. We refer to a deterministic stopping time (the special case where τ is chosen to be constant) as a *pre-committed* strategy, and shall see below that pre-committed strategies are not optimal in general.

We also classify errors, as follows:

1) If our prediction is $d = 0$ (stable) when the contingency is actually unstable (of any type), we say that *error* A_1 *occurs*,
2) If our prediction is $d > 0$ (unstable) when the contingency is actually stable, we say that *error* A_2 *occurs*,
3) If our prediction is $d > 0$ when the contingency is actually unstable but of a different instability type, we say that *error* A_3 *occurs*.

In this work we assume that a random sample of n contingencies is available, generated for example by a simulation procedure. For each contingency we assume that the rotor angle vectors $\delta(t)$ are given at discrete times $t_1 < t_2 < \cdots < t_M$, along with the contingency's stability class $C \in \mathcal{C}$. We take the uniform probability measure, denoted by $\mathbb{P}$, on this dataset. Thus the probability of error A_i for a given decision rule and dataset is interpreted simply as the proportion of contingencies in the dataset for which error A_i occurs under that decision rule. Equivalently we compute expectations as simple averages over the contingencies in the dataset. Taking values $p_i \in [0, 1], i = 1, 2, 3$ and letting $\mathbb{E}$ denote expectation, the problem is then:

(RCP) Risk-constrained sequential testing problem: *Choose the pair of decision variables* (τ, d) *in order to minimise the average time* $\mathbb{E}[\tau]$ *taken to make the prediction* d*, subject to the following risk constraints:*

$$\mathbb{P}[A_1] \leq p_1, \tag{1}$$
$$\mathbb{P}[A_2] \leq p_2, \tag{2}$$
$$\mathbb{P}[A_3] \leq p_3. \tag{3}$$

III. STRUCTURE OF OPTIMAL DECISION RULE

This version of the classification problem is closely related to the class of sequential testing problems studied for example in Chapter 6 of [15]. As in the latter reference, our approach will be to identify *optimal boundaries*. These boundaries may be used in an online fashion to detect the stopping time τ. In particular we construct a process $\pi^0 = (\pi^0(t))_{t_1 \leq t \leq t_M}$, a lower boundary $\ell : [t_1, t_M] \to [0, 1]$ and an upper boundary $u : [t_1, t_M] \to [0, 1]$. The value $\pi^0(t)$ may be interpreted in a Bayesian sense as the posterior probability

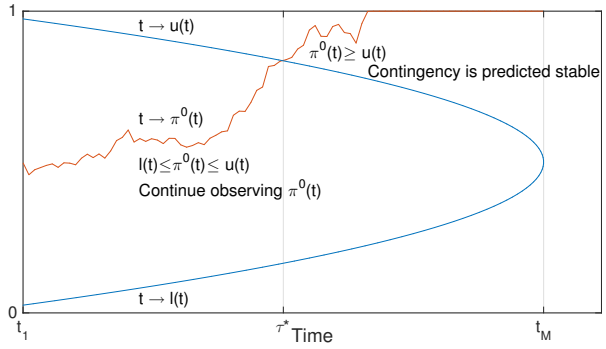

Fig. 1. A stylised illustration of the lower boundary $\ell(t)$, upper boundary $u(t)$, and trajectory of the process π^0.

that the contingency is stable given the observation at time t. The boundaries u and ℓ are functions of time and satisfy the constraint $0 \leq \ell(t) \leq u(t) \leq 1$ for each t, together with the terminal constraint $\ell(t_M) = u(t_M)$. The decision variables (τ, d) are then constructed as follows:

(i) The stopping time τ is the first time that the process π^0 satisfies either $\pi^0(t) \geq u(t)$ or $\pi^0(t) \leq \ell(t)$. Thus if $\ell(t_1) < \pi^0(t_1) < u(t_1)$ as in Figure 1, then τ is the first time that the process π^0 crosses either boundary u or ℓ,

(ii) If π^0 crosses the upper boundary first (or if the initial value $\pi^0(t_1) \geq u(t_1)$) then we immediately predict $d = 0$, i.e. that the contingency is stable,

(iii) if π^0 crosses the lower boundary first (or if the initial value $\pi^0(t_1) \leq l(t_1)$) then we immediately predict that the contingency is unstable, predicting the particular instability class d which appears most likely given the observations at time τ.

In this construction the constraint $0 \leq \ell(t) \leq u(t) \leq 1$ for each t is thus natural. The stopping time τ is detected 'just in time', in the sense that the prediction d is made immediately upon detection of the boundary crossing. The terminal constraint $\ell(t_M) = u(t_M)$ ensures that a prediction is always made by the terminal time T. In the illustration of Fig. 1 the process π^0 crosses the upper boundary first, at the time τ^*. In this case we would therefore make the prediction at time τ^*, declaring that $d = 0$.

In contrast to the treatment in [15], however, our probability measure $\mathbb{P}$ is empirical. In the present work we therefore do not seek analytic solutions but instead aim to *learn* the necessary posterior probabilities using PNNs, as described in Section IV. However we first make the risk-constrained problem **(RCP)** more tractable by reformulating it as an unconstrained optimisation problem via the Lagrange method with inequality constraints.

A. Lagrangian minimisation problem

Denote by υ the pair of boundaries

$$\upsilon = \{(\ell(s), u(s)) : t_1 \leq s \leq t_M\}. \tag{4}$$

Given these boundaries υ we may construct the decision rule $(\tau_\upsilon, d_\upsilon)$ as described in Section III. For each contingency i

in a given dataset, the declared stability class d_υ^i may then be compared with its true stability class. In this way we may determine whether or not the decision rule $(\tau_\upsilon, d_\upsilon)$ gives rise to error A_i for each $i = 1, 2, 3$ and for each contingency in the dataset. We then call the boundaries υ *feasible* if they satisfy the risk constraints (1)-(3). Note that these constraints can be written as $\mathbb{E}[\mathbf{1}_{A_i}] - p_i \leq 0$, for $i = 1, 2, 3$. Here, for each contingency i the *indicator function* $\mathbf{1}_{A_i}$ takes the value 1 if event A_i occurs in that contingency and takes the value 0 otherwise. To each of these inequalities let us assign a Lagrange multiplier $\lambda_i \geq 0$, and write the corresponding Lagrangian function

$$L(\upsilon; \lambda) = \mathbb{E}\left[\tau_\upsilon + \sum_{i=1}^{3} \lambda_i(\mathbf{1}_{A_i} - p_i)\right], \tag{5}$$

where $\lambda = (\lambda_1, \lambda_2, \lambda_3)$ is the vector of multipliers.

By Proposition 3.3.4 of [18] a solution of the following unconstrained problem also solves the original constrained problem:

(UP) Unconstrained problem:

Find a nonnegative multiplier vector λ^* and feasible boundaries υ^* such that

$$\lambda_i^* \mathbb{E}[\mathbf{1}_{A_i} - p_i] = 0, \qquad \textit{for } i = 1, 2, 3, \tag{6}$$

and such that the boundaries υ^* minimise the Lagrangian, so that

$$L(\upsilon^*; \lambda^*) = \inf_{\upsilon} L(\upsilon; \lambda^*). \tag{7}$$

In the next section we provide details of our approach to solving the unconstrained problem **(UP)**.

IV. NUMERICAL SOLUTION

We assume that our sample of n contingencies is divided into three disjoint subsamples which we will call the *PNN training*, *boundary training*, and *test* data, containing respectively n_P, n_B and n_S contingencies, where $n_P + n_B + n_S = n$. For each time t and sample number $i \in \{1, \ldots, n_P\}$ in the PNN training data, let us write C_P^i and $\delta_P^i(t)$ for its stability class and its vector of rotor angles at time t respectively; define C_B^i, $\delta_B^i(t)$, C_S^i and $\delta_S^i(t)$ analogously.

A. Role of the PNN

Firstly for each observation time t_m, $m = 1, \ldots, M$, a PNN is created whose input is the vectors $\delta_P^1(t_m), \ldots, \delta_P^{n_P}(t_m)$ and whose target vector is $(C_P^1, \ldots, C_P^{n_P})$ (note that this is independent of m). Thus for each observation time t_m a probabilistic neural network, which we denote PNN_m, is trained to learn configurations of rotor angles that are characteristic of the different stability classes. More precisely, let $\hat{\delta}$ be any test vector of rotor angles observed at time t_m. Then PNN_m maps $\hat{\delta}$ to a discrete probability distribution: that is, a nonnegative vector

$$\pi = (\pi^0, \pi^1, \ldots, \pi^I) \tag{8}$$

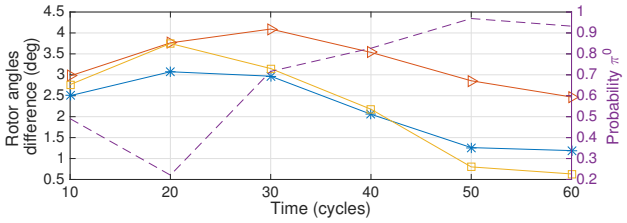

Fig. 2. Illustrative training trajectories for the probabilistic neural networks in the two-rotor example of Section IV-A. The difference between the two rotor angles is plotted for the unstable training trajectory (red, triangular markers) and the stable training trajectory (blue, asterisks). A test trajectory (yellow, squares) is also shown, together with illustrative output from the trained PNNs (purple dashes).

with $\sum_{j=0}^{I} \pi^j = 1$ and where π^j is a measure of distance from $\hat{\delta}$ to the jth stability class. This may be interpreted as the posterior distribution of the stability class $j \in \{0, 1, ..., I\}$, given that the vector $\hat{\delta}$ of rotor angles is observed at time t_m. The idea is illustrated in Fig. 2 in the following two-rotor example.

Example. In order to illustrate the role played by PNNs in our approach, let us temporarily consider only the binary classification problem (that is, distinguishing stability from instability) in a system with two generators. We take the smallest nontrivial PNN training dataset, namely $n_P = 2$ with the first contingency ($i = 1$) labelled as stable and the second ($i = 2$) labelled as unstable. Suppose that $(t_1, \ldots, t_M) = (10, 20, \ldots, 60)$ cycles. This means that 6 PNNs are trained: PNN_m is trained using the two vectors $\delta_P^1(10m)$ and $\delta_P^2(10m)$, where the vector $\delta_P^i(10m)$ consists of the two generator rotor angles at time $10m$ cycles in the ith PNN training sample. The trajectory of differences between these rotor angles is plotted in Fig. 2. In order to illustrate the output of the trained PNNs we also plot the trajectory of rotor angle differences for a test contingency, and the resulting PNN outputs $\pi^0(10), \ldots, \pi^0(60)$. It can be seen that at time 10 cycles, the posterior probabilities are equal (0.5) for both the stable and unstable class. The posterior probability then favours the unstable class at time 20 cycles, then consistently favours the stable class from 30 cycles onwards. This completes our example.

Next the trained PNNs are applied to each sample in the boundary training data in turn, thus mapping trajectories of rotor angles into trajectories of posterior distributions. More precisely, taking the ith sample $(\delta_B^i(t_1), \ldots, \delta_B^i(t_M))$ in the boundary training data and applying the PNNs, we obtain a time series $\pi_i = (\pi_i(t_1), \pi_i(t_2), \ldots, \pi_i(t_M))$, where $\pi_i(t_m)$ is the posterior distribution of the stability class at time t_m. These n_B time series will be used together with dynamic programming in Section IV-B to construct the optimal boundaries.

Each sample in the test dataset is similarly converted from a trajectory of rotor angles into a trajectory of posterior distributions by applying the trained PNNs. For each test sample, this allows the associated optimal stopping time τ to be detected and the stability prediction d made, as

described in (i)–(iii) of Section III. It remains to make precise the step (iii). If the lower boundary is crossed before the upper (at time τ) then we immediately inspect the posterior probabilities $(\pi_i^1(\tau), \pi_i^2(\tau), \ldots, \pi_i^I(\tau))$. The maximum of these posterior probabilities corresponds to the most likely unstable class $d \in \{1, \ldots, I\}$ given the rotor angles observed at time τ (conditional on the contingency being unstable), and this unstable class is then immediately declared.

B. Computing optimal boundaries

The optimal boundaries are obtained as follows, using the boundary training dataset. For each given multiplier vector λ a backward induction procedure is employed from $t = t_M$ to $t = t_1$. In this procedure we consider a sequence of subproblems of **(UP)** in which the optimisation begins at some fixed time $t \leq t_M$. Correspondingly we generalise definition (4), denoting by $\upsilon(t)$ the portion of the boundaries given by

$$\upsilon(t) = \{(\ell(s), u(s)) : t \leq s \leq t_M\}. \tag{9}$$

We also generalise the Lagrangian (5) to

$$L(t, \upsilon(t); \lambda) = \mathbb{E}\left[\tau_\upsilon - t + \sum_{i=1}^{3} \lambda_i(\mathbf{1}_{A_i} - p_i)\right]. \tag{10}$$

Fixing λ, at each step the Lagrangian $L(t, \upsilon(t); \lambda)$ is minimised over the choice of boundaries $\upsilon(t)$. As t_M is the terminal time, in this case one must stop immediately and make a decision. Hence the boundaries ℓ and u coincide at t_M, i.e. we impose $\ell(t_M) = u(t_M) = x$ for some $x \in [0, 1]$, making the Lagrangian a one dimensional function of x. Therefore it is enough to minimise the Lagrangian $L(t_M, x; \lambda)$ over x and set $\ell(t_M) = u(t_M) = x^*$, where x^* is the minimiser.

For $t = t_j$ with $j < M$, suppose inductively that we know the optimal boundaries $\ell^*(t), u^*(t)$ for each $t = t_{j+1}, \ldots, t_M$. We now compute the optimal boundaries at time t_j. Note that now the Lagrangian $L(t_j, \upsilon(t_j); \lambda)$ is a two-dimensional function of $(x, y) = (\ell(t_j), u(t_j)) \in [0, 1] \times [0, 1]$ with $x \leq y$. By minimising $L(t_j, \upsilon(t_j); \lambda)$ with respect to (x, y) we get $\ell^*(t_j) = x^*$ and $u^*(t_j) = y^*$, where (x^*, y^*) is the minimiser. The backward induction step is repeated until $t = t_1$ to obtain the optimal boundaries ℓ^* and u^* over $[t_1, t_M]$.

In this way, for each nonnegative vector λ of Lagrange multipliers the optimal boundaries υ_λ^* are obtained. We exclude any λ which leads to infeasible optimal boundaries υ^* by setting $L(\upsilon^*; \lambda) = \infty$ for such λ. Finally we minimise the Lagrangian $L(\upsilon_\lambda^*; \lambda)$ over λ to obtain a candidate solution to **(UP)**. Note that in the empirical approach of this paper, the quantities p_i and $\mathbb{E}[\mathbf{1}_{A_i}]$ take values on different lattices in general and as such, the condition (6) can only be expected to hold approximately. Instead these values are reported for our case study in Sections V and VI-A respectively.

V. CASE STUDY

To illustrate the potential of the proposed methodology we apply it to a 16-machine, 68-bus, five-area network representing a reduced order equivalent model of the interconnected New England Test System and New York Power System as detailed in [20] and [21]. The 16 generators are represented by sixth order models. Generators 1-8 are equipped with a slow IEEE-DC1A excitation system, while generator 9 uses a fast acting IEEE-ST1A1 static exciter and power system stabiliser. The remaining generators 10-16 are under constant manual excitation control. Power system loads are modelled as constant impedance.

A PNN training dataset with 3000 contingencies and a boundary training dataset with 2000 contingencies were constructed with the same system data and specifications as those in [2] (see their Section III.A), with rotor angle responses given at times $t = 10, 20, \ldots, 60$ cycles and the convention that 60 cycles is equivalent to 1 second. We assume that generator 13 is slack and has a constant rotor angle. An additional test dataset with 2000 contingencies was also constructed. Applying the classification method of [2] to all 7000 simulations led to the identification of 14 different instability classes in total. The PNN training dataset had in total 2733 stable contingencies with the remaining 267 unstable contingencies distributed among 10 instability classes, while the boundary training dataset had 1833 stable contingencies and 14 instability classes were represented among the 167 unstable contingencies. The test dataset had 1829 stable contingencies and 12 instability classes were represented among the 171 unstable contingencies. The two instability classes not observed in the test dataset occurred once each in the training datasets.

A. Choice of error constraints p_i

The values p_i specify the target accuracy for the classification procedure. Since the decision rule is constructed by learning from the training datasets, it is clear that classification error rates equal to zero, for example, may not be achievable when applied to the test dataset. In particular, as described above, by chance there happen to be four instability classes which are absent from the PNN training dataset. The training datasets should therefore be suitably large, in order to provide a sufficiently faithful representation of the set of possible contingencies and their stability classes.

Our objective is therefore stated in relative terms: we impose risk constraints comparable to the accuracy achieved by the longest pre-committed observation times considered, and then minimise the average time taken to make the prediction. Since some trade-off between speed and accuracy may be expected, we will interpret this objective by prioritising the minimisation of error type A_1 over types A_2 and A_3. To choose the error rate bounds p_i we therefore aim to be as strict as possible with error A_1 while allowing some flexibility for error types A_2 and A_3 (these priorities could equally have been chosen differently).

Recall that the test dataset had 1829 stable and 171 unstable contingencies. From Table I the lowest frequency

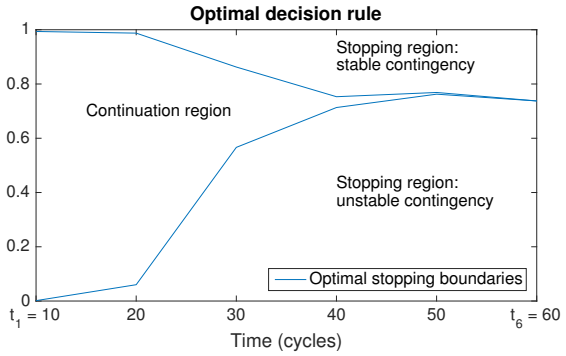

Fig. 3. Optimal upper and lower boundaries as function of time.

for error A_1 achieved by a pre-committed strategy on this dataset was $5/171$, for predictions always made at 40 cycles. In this case the remaining error rates were $2/1829$ for A_2 and $18/171$ for A_3. (The relatively high error rate for A_3 is due to several instability classes having a small number of representatives in the training datasets.) When these values were imposed as error rate bounds in the constrained problem, however, the algorithm failed to find feasible boundaries v. The risk constraints were therefore successively relaxed until a feasible optimal solution was found. For the results in this paper we then specified a maximum of six errors of type A_1 and A_2 by setting $p_1 = 6/171$ and $p_2 = 6/1829$, and a maximum of 22 errors of type A_3 by setting $p_3 = 22/171$.

VI. NUMERICAL RESULTS

For the results in this section, all PNNs were generated by the MATLAB function 'newpnn' [19], which is a two-layer network, without applying a neural transfer function to the output of the second layer.

A. Optimal boundaries

A pattern search algorithm [22] was employed to minimise the Lagrangian over λ, with an accuracy of 0.1 in each component λ_i. The optimisation algorithm was found to converge to a feasible solution with $(\lambda_1, \lambda_2, \lambda_3) = (310.9, 127.9, 0.0)$, whose units are cycles.

Fig. 3 displays the optimal lower and upper boundaries ℓ^* and u^* as functions of time. As time progresses and more information becomes available, we observe that the upper boundary tends to decrease while the lower boundary tends to increase, and consequently the continuation region (the region where we postpone making predictions) tends to narrow over time. (For completeness we also report the values $\mathbb{E}[\mathbf{1}_{A_i}]$, which are the error rates on the *boundary training* data which appear in condition (6), see the discussion at the end of Section IV. For $i = 1, 2, 3$ these are 6/167, 6/1833 and 21/167 respectively).

B. Distribution of optimal decision time

We see from Fig. 3 that by time $t = 40$ cycles (0.677s) the lower and upper boundaries are separated by a relatively small vertical distance (significantly less than 0.1). This suggests that the process π^0 will typically cross one or other boundary before time $t = 40$ cycles. In order to make this assertion precise we record, for each simulation in the test dataset, the decision time τ. Inspection of the results reveals that under our optimal decision rule most contingencies (89.95%) are classified at the earliest opportunity ($t = 10$ cycles, or 0.167s). A significant minority (7.85%) of contingencies are classified at $t = 20$ cycles (0.33s), while none are classified after $t = 50$ cycles (0.83s). From these results we find that the average decision time $\mathbb{E}[\tau]$ is 11.25 cycles (0.188s) under the optimal decision rule.

In our problem formulation, it is of course possible for the decision time to be longer than this average in the relatively rare unstable contingencies. While late detection of the stable contingencies may not represent a concern, it is the early detection of unstable contingencies which is necessary to avoid system instability. When the average decision time is calculated only over the unstable contingencies in the test data, however, it rises to just 12.14 cycles (0.2s) and our conclusions are qualitatively unchanged.

C. Actual error rates

By construction the error rates satisfy the probabilistic constraints given in Section V when the optimal policy is applied to the boundary training data. Since both the boundary training and test data are merely samples, however, these error rates are of course not guaranteed to hold empirically when applied to the out-of-sample test data. Indeed the actual error rates for the optimal decision rule on our test data were (with the probabilistic constraints in brackets):

- Error A_1: 7/171 (6/171)
- Error A_2: 7/1829 (6/1829)
- Error A_3: 22/171 (22/171).

Nevertheless these actual error rates are only slightly larger than their corresponding error rate bounds. The discrepancy arises because the training and test datasets have different statistical properties such as the number of stable/unstable contingencies, instability classes and so forth. This effect may be reduced by ensuring that all datasets are sufficiently large. These error rates for our optimal decision rule are given in the 'Optimal' row of Table I.

D. Comparison with pre-committed prediction times

The main innovation in the present paper, relative to the existing literature on the identification of power system dynamic signatures, is that we allow the prediction time τ to vary. Thus when the initial observations are sufficiently informative, an appropriate prediction may be made without delay. Conversely in cases where the initial observations are uninformative, rather than making a quick but inappropriate declaration, the prediction is delayed until sufficient information is revealed. In this way we are able to derive a flexible decision time which is fastest on average subject to given risk constraints. We therefore close our analysis with a comparison to pre-committed prediction times.

For each of the deterministic prediction times $\tau = 10, 20, \ldots, 60$ cycles we constructed a PNN as above on our PNN training dataset. These PNNs were then used

TABLE I
ERROR RATES WITH PRE-COMMITTED STRATEGY AND SEQUENTIAL TESTING SOLUTION

Time (cycles)	Error rate A_1 (out of 171)	Error rate A_2 (out of 1829)	Error rate A_3 (out of 171)
10	25	5	19
20	13	5	19
30	9	2	18
40	5	2	18
50	6	3	19
60	6	3	19
Optimal (avg.):11.25	7	7	22

for classification at these fixed times. Table I provides the resulting error rates when applied to the test dataset. Only the pre-committed strategies for the later times $\tau = 40, 50, 60$ are within the probabilistic constraints, meaning that pre-committed strategies needed a relatively long observation time to meet these risk constraints. In contrast, as described in Sections VI-B and VI-C above, our optimal rule returned a classification within 11.25 cycles on average (or 12.14 cycles on average among the unstable contingencies only), nevertheless achieving an accuracy almost equal to that of these longer pre-committed times.

VII. CONCLUSIONS

In order to predict post-fault transient stability status as quickly as possible under risk constraints we have combined machine learning, in particular probabilistic neural networks, with statistical sequential analysis. Thus we allow the prediction time to vary, and achieve shorter average times while maintaining the prediction accuracy associated with longer pre-committed times. The optimal decision rules are computed via dynamic programming, as time-dependent boundaries for an online process of posterior predictive distributions derived from PNNs. The prediction is made at the first time these boundaries are crossed, and classification is then performed using the posterior distributions.

The accuracy and advantages of the method were demonstrated in simulations of the interconnected New England Test System and New York Power System. In this example predictions were made 0.188 seconds after the disturbance on average. The error rates for this optimal decision rule were comparable to those achieved by pre-committed prediction times of 0.5–0.67 seconds.

VIII. ACKNOWLEDGMENTS

This work was supported by the UK Engineering and Physical Sciences Research Council (EPSRC) grant EP/I031650/1. JM was supported by EPSRC grant EP/K00557X/2.

REFERENCES

[1] S. Rovnyak, S. Kretsinger, J. Thorp, and D. Brown, "Decision trees for real-time transient stability prediction," *IEEE Transactions on Power Systems*, vol. 9, no. 3, pp. 1417–1426, Aug. 1994.

[2] T. Guo and J. Milanovic, "Online Identification of Power System Dynamic Signature Using PMU Measurements and Data Mining," *IEEE Transactions on Power Systems*, vol. PP, no. 99, pp. 1–9, 2015.

[3] L. Wehenkel, T. V. Cutsem, and M. Ribbens-Pavella, "An artificial intelligence framework for online transient stability assessment of power systems," *IEEE Transactions on Power Systems*, vol. 4, no. 2, pp. 789–800, May 1989.

[4] N. Senroy, G. T. Heydt, and V. Vittal, "Decision Tree Assisted Controlled Islanding," *IEEE Transactions on Power Systems*, vol. 21, no. 4, pp. 1790–1797, Nov. 2006.

[5] M. He, V. Vittal, and J. Zhang, "Online dynamic security assessment with missing pmu measurements: A data mining approach," *IEEE Transactions on Power Systems*, vol. 28, no. 2, pp. 1969–1977, May 2013.

[6] M. He, J. Zhang, and V. Vittal, "Robust Online Dynamic Security Assessment Using Adaptive Ensemble Decision-Tree Learning," *IEEE Transactions on Power Systems*, vol. 28, no. 4, pp. 4089–4098, Nov. 2013.

[7] L. S. Moulin, A. P. A. d. Silva, M. A. El-Sharkawi, and R. J. Marks, "Support vector machines for transient stability analysis of large-scale power systems," *IEEE Transactions on Power Systems*, vol. 19, no. 2, pp. 818–825, May 2004.

[8] A. D. Rajapakse, F. Gomez, K. Nanayakkara, P. A. Crossley, and V. V. Terzija, "Rotor Angle Instability Prediction Using Post-Disturbance Voltage Trajectories," *IEEE Transactions on Power Systems*, vol. 25, no. 2, pp. 947–956, May 2010.

[9] F. R. Gomez, A. D. Rajapakse, U. D. Annakkage, and I. T. Fernando, "Support Vector Machine-Based Algorithm for Post-Fault Transient Stability Status Prediction Using Synchronized Measurements," *IEEE Transactions on Power Systems*, vol. 26, no. 3, pp. 1474–1483, Aug. 2011.

[10] C.-W. Liu, M.-c. Su, S.-S. Tsay, and Y.-J. Wang, "Application of a novel fuzzy neural network to real-time transient stability swings prediction based on synchronized phasor measurements," *IEEE Transactions on Power Systems*, vol. 14, no. 2, pp. 685–692, May 1999.

[11] N. I. A. Wahab, A. Mohamed, and A. Hussain, "Fast transient stability assessment of large power system using probabilistic neural network with feature reduction techniques," *Expert Systems with Applications*, vol. 38, no. 9, pp. 11 112–11 119, 2011.

[12] A. G. Bahbah and A. A. Girgis, "New method for generators' angles and angular velocities prediction for transient stability assessment of multimachine power systems using recurrent artificial neural network," *IEEE Transactions on Power Systems*, vol. 19, no. 2, pp. 1015–1022, May 2004.

[13] N. Amjady and S. F. Majedi, "Transient Stability Prediction by a Hybrid Intelligent System," *IEEE Transactions on Power Systems*, vol. 22, no. 3, pp. 1275–1283, Aug. 2007.

[14] S. Kretsinger, S. Rovnyak, D. Brown, and J. Thorp, "Parallel decision trees for predicting groups of unstable generators from synchronized phasor measurements," in *Precise Measurements in Power Systems Conference, Arlington, Virginia*, 1993.

[15] G. Peskir and A. Shiryaev, *Optimal Stopping and Free-Boundary Problems*. Springer Science & Business Media, Nov. 2006.

[16] D. F. Specht, "Probabilistic neural networks for classification, mapping, or associative memory," in *Neural Networks, 1988., IEEE International Conference on*. IEEE, 1988, pp. 525–532.

[17] L. Rutkowski, "Introduction to Probabilistic Neural Networks," in *New Soft Computing Techniques for System Modeling, Pattern Classification and Image Processing*, ser. Studies in Fuzziness and Soft Computing. Springer Berlin Heidelberg, 2004, no. 143, pp. 21–57.

[18] D. P. Bertsekas, *Nonlinear programming*. Athena Scientific, 1999.

[19] P. D. Wasserman, *Advanced Methods in Neural Computing*, 1st ed. New York, NY, USA: John Wiley & Sons, Inc., 1993.

[20] G. Rogers, *Power System Oscillations*. Springer Science & Business Media, Dec. 2012.

[21] B. Pal and B. Chaudhuri, *Robust control in power systems*. Springer Science & Business Media, 2006.

[22] R. Lewis and V. Torczon, "A Globally Convergent Augmented Lagrangian Pattern Search Algorithm for Optimization with General Constraints and Simple Bounds," *SIAM J. Optim.*, vol. 12, no. 4, pp. 1075–1089, Jan. 2002.