ABSTRACT
Model Predictive Control (MPC) uses a system model and optimisation algorithm within the controller, calculating the best set-points at each time step given the current and predicted weather, occupancy, and other conditions. It is an established technique in other fields and has been receiving growing attention by buildings researchers, but has yet to find its way into common practice in this field. This paper presents a modified approach that uses pre-calculation of optimal set-points over a grid of possible conditions and interpolation of the resulting lookup table for real-time control. Methodological details are considered and the range of applicability and relative performance is discussed, referencing case studies. Open-source software for this approach is also presented. It is hoped that the methods and software will provide researchers, designers and operators with a practical way to devise better supervisory controls for low-energy buildings.

INTRODUCTION
With a growing demand for more energy efficient buildings, designers are being pushed to integrate previously independent systems such as HVAC and solar shading, to make use of site and program peculiarities like local heat sources and sinks, and to use lower power density HVAC systems such as radiant cooling and natural ventilation. Intelligent operation of these more complex and integrated systems is often crucial to their success; designing control strategies must be an integrated part of the design process. But designing good controls for such systems is challenging.

Online MPC
Online MPC offers a way of approaching such problems. Instead of trying to define the control logic explicitly, a building model and an optimisation algorithm are used within the control system in real-time to calculate the best setpoint values given the current and predicted conditions. At each controller time step, an optimal sequence of control values over a prediction horizon is calculated, only the first of which is implemented, and at the next controller time step the horizon shifts forward one step and the process is repeated. (Note that a number of variants on this are also possible, such as implementing two or more time steps of inputs and performing the optimisation less often, or having different control and prediction horizon lengths.) MPC is widely used in other fields (Qin and Badgwell (2003) notes its use in more than 4,000 industrial applications). It is well suited to the control of non-linear systems with strong state constraints but relatively slow system dynamics, which is often found in chemical process control. It was a proven practical technique before it was studied theoretically. Good overviews of the field are available in Morari and Lee (1999) and Mayne et al. (2000). There has also been a growing number of MPC studies for building systems over the past decade, stemming not from controls research but from building energy simulation research. (See for example Mahdavi (2001), Clarke et al. (2002), Henze et al. (2005), and see the review in Coffey et al. (2010). The paper by Clarke et al also compares MPC to fuzzy logic and neural network controllers, noting the various benefits of having a physics-based model in the controller, including the avoidance of a learning period.) Some more recent work has come from controls researchers from other fields turning their attention to buildings (e.g. Ma et al. (2010) and Oldewurtel et al. (2010)). Potential for energy savings, demand reduction and performance improvement has been shown with a wide variety of systems, including chilled water storage, radiant slab pre-cooling and integrated HVAC and facade control. And as buildings become more complex the benefits of MPC are expected to become more pronounced.

But MPC is currently far from common practice in building design and operation. It is difficult to use most building simulation tools for this because of their slow run-times and the fact that many do not allow the user to explicitly specify initial state values, and the software used by most controls researchers is unfamiliar to most buildings researchers and practitioners. In addition, online optimisation is difficult to implement with existing building control systems, and the fact that the control logic is implicit rather than explicit makes it difficult for system designers to integrate it into their design processes.

Offline optimization
For some types of MPC problems, multiparametric programming can be used to solve the problem explicitly, providing a set of control laws that fully cover the conditions space and that exactly replicate con-
control behaviour of online MPC (Bemporad et al., 2002). However, this can only be used with certain types of MPC problems (e.g. linear or switched-linear models with linear or switched-linear objective functions), into which most of the challenging building control problems do not fit. And with the possible future exception of Modelica (Wetter, 2009b), this approach would not be possible with any of the commonly used building simulation tools.

But the idea of explicit MPC is very appealing for buildings applications, because it would be easier to implement in existing building control systems than online MPC, and being able to visualize optimal control responses over the full conditions space could provide useful feedback during system design. Finding ways to approximate it using common building simulation tools could thus be worthwhile, even if it brings with it some performance penalties relative to online MPC. Current work by May-Ostendorp and Henze considers the approach of simulating online MPC over some or all of a representative weather year and then using statistical techniques to derive near-optimal control laws from the results. This could provide a useful way of getting these benefits. The research presented herein (and in more detail in (Coffey, 2011)), considers the following approach: define a grid of conditions (initial states and predicted disturbances) that covers the range of what the system will face, solve the MPC optimisation problem at each point in the grid, and then use the resulting grid of optimal control responses as an interpolation lookup table in real-time control.

METHODS

Offline optimisation over a grid of conditions

Open-source software for MPC with standard building simulation tools was developed in previous research (Coffey et al., 2010), using GenOpt (Wetter, 2009a) as the optimizer, which allows for the use of any text-file based building simulation tool that can be called from the command line. Figure 1 shows the same basic structure being used to calculate control lookup tables.

Given a model of the building and/or system of interest, the steps involved in using this process are:

- Create template input files by demarcating the control variables (with %) and conditions variables (with $) in the model and weather files, and locate the objective function in the output file.
- Define bounds and precisions for the control variables, and define conditions variables to be used in the grid (this may be a smaller set than the conditions variables used in the model, as discussed below) along with bounds and spacings for the grid. Configure GenOpt and SimCon with these.
- Use SimCon and GenOpt to solve for the optimal control values at each point in the conditions grid. This is usually a computationally expensive process, but is easily parallelized, and with the use of many virtual machines on now easily-accessible cloud computing platforms, the question is more about money than about time.
- The resulting grid of optimal control values can then be used as an interpolation lookup table for control in simulated or physical implementations, and the multidimensional grid can be visualized through scatter plots or by graphing 2- or 3-dimensional slices through it, providing important feedback to the design process.
- This process can be used iteratively. It is wise to start by solving with a very coarse grid, and possibly with fewer conditions or control variables, fixing model bugs or adjusting objectives and bounds based on the feedback from the grid visualizations, and then building up the precision of the lookup table over 3 or 4 (or possibly more) iterations, keeping grid point solutions from the previous iteration if no changes were made.

Range of applicability

The basis for Figure 2 below is simply that the computation time required for lookup table creation is the product of the conditions grid size and the average computation time per grid point. The dollar costs are based on $0.10 / processor-hour, which is roughly the current commercial cloud computing cost for small-scale users. The shaded area is a conservative cut-off for financial feasibility for a consulting or design firm working on a single building, assuming some iteration in the process. The figure highlights the trade-offs between model complexity, optimisation precision and grid spacing, and shows the scale of problems that are feasible - simulation time must be within seconds (note that this is over a simulation horizon of hours or days, rather than a full year), and the number of conditions variables must be kept to less than roughly 5 or 6. As such, this approach usually requires approximations to limit the dimensionality of the lookup table. Much of the research described herein is aimed at better understanding the performance costs associated with various approximation techniques.
Approximation techniques

Consider a control problem that requires day-ahead predictions of ambient temperatures, such as the overnight cooling of a massive chilled floor or ceiling. If hourly predictions were used, this would require 24 dimensions in the conditions grid just for the ambient temperature, making the offline solution computationally infeasible. One way of decreasing the dimensionality is to use a coarser prediction, for example using average temperatures for 4-hour blocks instead of 1-hour blocks. Another approach is to take advantage of the expected shape of the curve, and use a small number of parameters to define the prediction, such as the maximum and minimum temperatures and maybe the time of their occurrence. In any such variation, a normalized curve is required to produce the values of the temperature at each timestep of the simulation, based on the predicted values of the parameters. Normalized curves can be derived using typical or historical data for the site or system under consideration, as demonstrated below. Similar approaches can be used for other disturbance variables.

Even with the approximations of input parametrization, many control problems in buildings still have too many dimensions to be tractable as lookup tables. However, the approach may still be useful in such a case, if the structure of the problem allows it to be decomposed into a hierarchical set of problems where some of the subproblems are small enough to be solved offline. This is discussed in detail and used in case studies in (Coffey, 2011).

Open-source software to facilitate approach

The SimCon beta software described in (Coffey et al., 2010) has been extended to be used for this offline approach. It is written in java, and the source code is freely available for download (site tdb). The software’s current functionality includes: the option of running in either online MPC mode or in lookup table calculation mode, so the same software can still be used for online MPC, including cases with decomposed problems that involve a higher-level problem that must be solved online and lower-level problems that can be solved offline as lookup tables; core methods to set up a sequence of optimisation problems for a user-defined conditions grid, solve them using GenOpt, and record the results to a lookup table file; multi-variable interpolation for lookup tables stored in a text file; and an extensible library of algorithms to convert conditions inputs to and from parametrized forms. Peripheral components are also included in the open source code, including sample files for running annual simulations with the controller in the Building Control Virtual Test Bed (BCVTB, Wetter and Haves (2008)), java methods for collecting weather predictions from the National Digital Forecast Database, sample java-based interfaces for human-in-the-loop implementations, and a visualization tool for the lookup tables and a weather conditions parametrization tool, both currently in Excel-VBA.

ILLUSTRATIVE CASE STUDY

This simple case study illustrates the basics of the procedure. Because the problem is small, no problem decomposition or conditions parametrization is required. Consider a single south-facing zone with automated external shading and automated natural ventilation. A supervisory controller must determine hourly setpoints for the shading percentage (defined as the percentage decrease in solar gains, not necessarily a linear function of the shade angle) and natural ventilation percentage (defined as the number of air changes per hour relative to minimum and maximum possible values) to minimize the combined heating, cooling and lighting energy consumption. To keep the problem simple, the zone is considered as massless, glare is not considered in the shading control, a highly simplified approach is used for the daylighting calculation, the artificial lights are assumed to dim perfectly to maintain a prespecified lighting level, the daylighting and solar gains vary in unison when the shading percentage varies, and the variable capacity for natural ventilation under different wind and thermal conditions is ignored.

The model was written in Modelica. The conditions grid used is as shown in Table 1, which results in 2880 sets of disturbances for which optimal control must be determined. The Hookes-Jeeves algorithm in GenOpt was used, with 5 step size reductions, and non-parallelized so running on just one processor. The sequential optimisations required approximately 84.5 hours (3.5 days) on a single Windows virtual machine.

<table>
<thead>
<tr>
<th>Table 1: Conditions grid for shading and nat vent</th>
</tr>
</thead>
<tbody>
<tr>
<td>ambient temp (C)</td>
</tr>
<tr>
<td>direct solar (W/m² on window)</td>
</tr>
<tr>
<td>diffuse solar (W/m² on window)</td>
</tr>
<tr>
<td>gains from people and plugs (W/m² on floor)</td>
</tr>
</tbody>
</table>
Figures 3 and 4 show slices through the resulting control lookup table. Figure 3 shows the optimal shading and natural ventilation levels as a function of the ambient temperature, direct and diffuse solar irradiation, and internal heat gains. Figure 4 highlights the response of the optimal shading value to different conditions values for direct and diffuse solar. The results make intuitive sense: the natural ventilation level increase from 0% to 100% as ambient temperature goes from cold up to the zone temperature setpoint and then drops off to 0%; the solar shading level increases with higher solar gains, except when the ambient temperature and internal loads are low enough that zone requires heating; at low ambient temperature conditions, the solar shading only approaches zero as the solar gains approach zero, but when the ambient temperature is high, the shading control blocks the sun completely when the direct solar is high and the diffuse solar is low (cooling these solar gains is more costly than using the artificial lights), but with a more favorable direct-diffuse split the shading control allows just enough daylighting in to eliminate the artificial lighting requirement.

**Annual simulations**

To test the lookup table control, an annual simulation was run using TMY data for San Francisco. The TRNSYS weather reader was used to calculate the direct and diffuse solar gains on the window for each hour of the year, and a weekly schedule of internal loads was created for the simulation. The annual simulations were carried out using a copy of the Modelica model as the ‘real’ system, connected to a controller through the BCVTB, assuming no model mismatch and perfect disturbance predictions. A full annual simulation with hourly calls to the lookup table controller requires about an hour of computation. For comparison with the lookup table controller, a base case controller was also simulated, with control rules shown in grey in Figure 6. Note that the base case controller is a heuristic simplification of the lookup table controller by considering only one condition variable, and is intended only as a simple comparison point and to illustrate the importance of considering all of the interactions; controlling the blinds with a daylight sensor and the operable windows as an extension of the HVAC system’s economizer would provide a much higher-performing
baseline over which the lookup table controller would likely find much less improvement (such a study may be considered in future work). Also for comparison, three two-week periods were simulated with full online MPC, using the same optimisation configuration used in the lookup table creation, but with the actual disturbance values for each hour of the simulation, connected again through BCVTB. A full-year online MPC simulation was not attempted because of the computation time requirements - each of the two week periods required 6-8 hours of run time, so a full year simulation would require roughly 150 to 200 hours (6-8 days). The three simulation periods capture summer solstice, winter solstice and a shoulder season.

Figure 5 shows the week of March 1-7. The lookup table control responses are very similar to the online MPC responses for all but a few hours during this period of the simulation. For this period, they are generally calling for both more shading and more natural ventilation than used by the base case control. This is likely because there are relatively high solar gains during this period, and the base control considers only ambient temperature in its decision, but the optimisation-based lookup table also considers the solar gains and so can make these adjustments. The scatter plots in Figure 6 show this perhaps more clearly, with the spread in the lookup controller points being caused by its response to different solar conditions and different internal loads. Also note in Figure 6 the drop in natural ventilation percentage in the lookup case between 20°C and 22°C. However, we know that the optimal control in this region, given our model configuration, should approach 100% natural ventilation as the ambient temperature approaches the zone temperature of 22°C from below, and then drop to 0% natural ventilation at exactly 22°C, as it does in the base case controller. The reason that it does not is because it is interpolating linearly between the grid points at 20°C and 22°C. This is an illustration of one of the performance penalties in the lookup table approach; a finer grid would decrease this penalty, and thus a trade-off must be made at some point between the cost of grid creation and the gains in performance.

The annual combined heating, cooling and lighting energy consumption during occupied hours is 54.1 kWh/m² in the base case and 33.4 kWh/m² in the lookup table control case, an annual savings of 38.2%. The three 2-week periods of simulation with the online MPC provides a comparison point on the other side of the coin, showing where the lookup controller is not performing as well an ideal controller (between 20°C and 22°C, for example). Over the six test weeks the lookup controller performed 49.6% better than the base case, while the online MPC performed 57.0% better; with the online MPC demarcating the maximum improvement over the base case, the lookup table control captured 87% of the available savings.

Figure 5: Control decisions and objective function outputs, March 1-14

Figure 6: Annual simulation, hourly control decisions versus ambient temperature
PARAMETRIZATION CASE STUDY

This case study illustrates the use of EnergyPlus within this procedure, as well as the use of conditions parametrization. The controller is asked to determine the start time and charging length for overnight charging of a massive chilled slab with a cooling tower only, in order to minimize the average percentage people dissatisfied (PPD) during the following day with a floating slab and zone temperature. A 48-hour prediction and control horizon is used for the ambient temperature, and the control time-step is 24 hours long. The EnergyPlus example file RadLoTempHydrCoolTower.idf (from the example files folder in the EnergyPlus version 6.0 standard release) is used for this case study. The one-story building has three zones, with a total floor area of 130 m². A hydronic low temperature radiant system is the only source of cooling, with the ‘chilled’ water supplied solely by a cooling tower. Only very minor changes were made to the example model file, to keep the case study simple and replicable: the slab heating system was turned off; the run period was changed to just four days; the HVAC control schedule variables were made into functions of the nightly charging lengths and start times; the hourly weather file values are overwritten as a function of the predictions; and the simulation outputs are simplified to output just the average occupied PPD for one zone of the building.

Conditions parametrization: disturbances

Just the daily maximum and minimum temperatures over the 48-hr prediction horizon are used as conditions for the controller. But EnergyPlus requires hourly inputs for temperature and many other weather variables, so a relationship must be derived to estimate these hourly values over the prediction horizon given the maximum and minimum temperatures. Figure 7 shows the daily temperature profiles for San Francisco (TMY3), with each coloured line representing one day. Figure 8 shows the daily temperature lines normalized to [1,0] by their maximum and minimum. The dotted black line is the hourly average of these daily normalized curves, and the solid black line is the [1,0] normalization of that dotted black line. The hourly values for this solid black line are used in the model to estimate the hourly temperatures over the day-ahead prediction given the predicted maximum and minimum temperatures. The other weather variables are normalized in the same way, and if their own daily maximum or minimum values show correspondence with the maximum or minimum temperature values, then this relationship and the normalization is used to fill in these hourly values for that variable. If no correspondence exists, then average hourly values are used.

Conditions parametrization: initial states

Similar approaches may also be used for initial state variables, if there are a lot of initial state values required in the model (e.g. in cases with thermal storage in many different tanks or massive building components). But the bigger concern pertaining to state initialization is that most building simulation tools do not provide easy access to the state variables: instead of being able to explicitly initialize states when using the tool for MPC, one often has to use an ‘initialization horizon’, simulating over hours or days or weeks of previous disturbances and control actions to get the model state values to be roughly equal to the observed initial state values. This means that instead of having a small vector of initial states as part of the conditions grid, there would have to be a much longer vector of previous disturbances and control actions in the grid, ballooning its dimensionality. Techniques must be used to avoid this problem when using such simulation tools. One approach is to define an initialization horizon for the simulation where all of the values are constant except some small number of parameters that can be modified to produce the desired initial states. As such, a desired initial state value can be translated into adjustable parameter values in the simulation tool. This approach was used in this case, for the initial slab surface temperature and initial zone air temperature, details of which are available in (Coffey, 2011). However, this work-around has proven difficult to implement in case studies involving more than one or two state variable of interest.
Table 2: Conditions grid for slab cooling study

<table>
<thead>
<tr>
<th>MIN</th>
<th>MAX</th>
<th>SPACING</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>26</td>
<td>2</td>
</tr>
<tr>
<td>2.0</td>
<td>3.5</td>
<td>0.5</td>
</tr>
<tr>
<td>16</td>
<td>32</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>4</td>
</tr>
<tr>
<td>16</td>
<td>32</td>
<td>8</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>6</td>
</tr>
</tbody>
</table>

Figure 9: Surface plots of optimal charging length and optimal ave PPD, with $T_{floor\,9pm} = 22^\circ C$, $T_{zone\,9pm} = 24^\circ C$, $T_{max\,Day2} = 16^\circ C$, $T_{range\,Day2} = 4^\circ C$

**Optimisation over the conditions grid**

The conditions grid is shown in Table 2. The 2880 optimisations (Hookes-Jeeves, 2 step size reductions) were carried out on 5 virtual machines on the Amazon Elastic Compute Cloud, and required approximately 275 processor-hours (11.5 processor-days). Two slices through the lookup table are shown in Figure 9. The optimal charge length is zero when predicted ambient temperatures and initial zone temperatures are low, ten hours when predicted ambient temperatures and initial zone temperatures are high, and it ramps in between.

**Annual simulations**

Annual simulations for San Francisco (TMY3) were run for the lookup table control and for two different base cases and three different online MPC configurations designed to tease apart performance losses due to grid spacing, disturbance parametrization, and initial state approximation. Figure 10 shows the daily charge lengths specified by the lookup table control and the base cases. (Analysis of the lookup table and annual simulation results shows that the PPD is not very sensitive to the start time, so this was fixed at 9:30pm in the base cases.) “Basecase1” does not use any day-ahead prediction and its charge length varies monthly. “Basecase2” was created after learning from the lookup table control outputs, and it represents a very good base case. The three online MPC cases are defined as follows: “MPC1” uses the same 6 conditions variables as the lookup controller, thus keeping the disturbance and initial state parametrizations, but runs online instead of interpolating and so defines how good the lookup table control could do if the grid spacing was zero; “MPC2” uses the actual hourly weather predictions rather than the parametrized approximations, and thus shows the performance gains that could not be captured because of the use of disturbance parametrizations; and “MPC3” uses a 7-day initialization horizon with historical weather and control values, and thus allows for an estimate of the performance loss due to initial state approximation. The results are summarized in Table 3. Defining the performance improvement potential as the difference between “MPC3” and “base-case1”, the lookup table control (even with its coarse gridding in this case) was able to capture 59% of the available performance improvement. “Basecase2”, a control rule derived from looking at the lookup table slices, was able to capture 40%. With a finer grid, the lookup table control could capture no more than 94% of the savings, with the remaining 6% being lost through the parametrizations.

**DISCUSSION**

The parametrization case study results suggest that the losses in performance are not so high to consider the approach as infeasible; further case studies are recommended. The consideration of adaptive grid sizing should be considered in future research.
The many details left out of this paper because of space restrictions can be found in (Coffey, 2011), along with more detailed case studies: a study of cogen dispatch with monthly demand charges, overnight charging of a chilled water storage tank for a campus, and integrated solar shading and HVAC control for a large office.

CONCLUSION

A method for near-optimal supervisory control using building simulation tools and offline optimisation is outlined herein. The approach is feasible for problems that can be suitably expressed as a function of roughly 5-6 conditions variables or fewer, or for problems that can be broken down into subproblems of this size. Conditions parametrization methods for limiting the problem dimensionality are described and a simulation-based case study suggests it can be used with just modest losses in performance.

The inability to explicitly specify initial states in most building simulation tools remains a problem. This should be fixed by simulation tool developers. In the meantime, for controls problems with one or two important state variables the work-arounds outlined in (Coffey, 2011) may be used, for larger problems users are recommended to use the subset of simulation tools that allow explicit state initialization.

Open-source software to facilitate the use of building simulation tools both in online MPC and in the modified approach described herein is available online. The promise of this software and approach is that it might fit well within existing design and operation processes, with visualization of control responses over the conditions space providing important feedback during the design process, and that the resulting controls could be easy to implement within existing building control systems. Further research, software interface development and early-adopter implementation is needed to move in that direction.

Designing good controls for low-energy building systems is challenging. It is hoped that the concepts and tools described herein will help designers and operators get closer to optimal performance.

ACKNOWLEDGEMENT

The research described herein was made possible by fellowships from the National Science and Engineering Research Council of Canada, the American Society of Heating Refrigeration and Air conditioning Engineers, and the University of California Berkeley, and through research projects at the Lawrence Berkeley National Laboratory. The author would like to thank Phil Haves, Gail Brager, Francesco Borrelli, Michael Wetter, Gregor Henze and Peter May-Ostendorp for their assistance and feedback.

REFERENCES

Bemporad, A., Borrelli, F., and Morari, M. 2002. Model predictive control based on linear program-

ming - the explicit solution. IEEE Transactions on Automatic Control, 47.


