

Lawrence Berkeley National Laboratory

LBL Publications

Title

From fault-detection to automated fault correction: A field study

Permalink

<https://escholarship.org/uc/item/34758298>

Authors

Pritoni, Marco

Lin, Guanjin

Chen, Yimin

et al.

Publication Date

2022-04-01

DOI

10.1016/j.buildenv.2022.108900

Copyright Information

This work is made available under the terms of a Creative Commons Attribution-NonCommercial License, available at <https://creativecommons.org/licenses/by-nc/4.0/>

Peer reviewed

31 **1. Introduction and background**

32 Buildings use 40% of primary energy globally, and account for 33% of direct and indirect carbon
33 emissions from fuel combustion (Economidou M, 2011). Based on an analysis of the most
34 common faults in building systems, studies estimate that the energy savings achievable from
35 correcting these faults ranges from 5 to 30% whole building savings (Fernandez, 2017, Roth et
36 al. 2005). Commercial fault detection and diagnostics (FDD) tools automate the process of
37 detecting faults and suboptimal performance of building systems and help to diagnose potential
38 causes (Dexter et al. 2001). They offer several interrelated benefits including energy savings and
39 improved operational efficiency, utility cost savings, persistence in savings over time, streamlining
40 operations and maintenance processes, and supporting continuous energy management
41 practices such as monitoring-based commissioning.

42 As buildings become more data rich, FDD technologies are increasingly adopted in commercial
43 buildings. There are over 30 full-featured FDD software product offerings in today's market in the
44 US and new software products continue to enter the market (Kramer et al., 2020). Building
45 operators at the forefront of technology adoption are using FDD to enable median whole-building
46 portfolio savings of 9% (Kramer et al., 2020). A FDD tool usually is a software layer on top of the
47 existing building automation systems (BAS). It integrates with BAS to obtain building system or
48 equipment operational data (e.g. temperature, pressure, flow rate). Extensive libraries of detection
49 logic are continuously run against the data, and results are presented through a graphical user
50 interface for resolution by operations and maintenance staff. A number of possible causes or
51 recommendations for correcting each fault are listed, requiring either additional data analysis by
52 the user or on-site inspection. In addition, tools may provide a report of the duration and frequency
53 of faults, cost and/or energy impacts, and relative priority levels (Granderson et al. 2017).

54 Although FDD tools are being used to enable cost-effective energy savings, there is a capability
55 gap. Today's FDD technologies operate in an open loop manner. Faults are identified by the FDD
56 tools, however, the identified faults must be corrected through manual human intervention
57 (Kramer et al., 2020). In practice, the need for human intervention to fix faults once they are
58 identified often results in delay or inaction, causing additional operations and maintenance costs
59 or deteriorating comfort conditions. This capability gap is not only technical, but also represents
60 market-relevant desired functionality on behalf of FDD users and technology providers
61 (Granderson et al., 2017). Therefore, realizing automated fault correction in commercial FDD
62 technology offerings closes the loop between passive diagnostics and active control, increase the
63 savings realized through the use of FDD tools, and reduce the extent to which savings are
64 dependent upon human intervention.

65 Kim and Katipamula (2018) indicate that since 2004, more than 100 FDD research studies
66 associated with building systems have been published. However, the academic publication has
67 extensively focused on the development of new FDD algorithms for HVAC systems (Katipamula
68 et al. 2005, Zhao et al. 2013, Wang et al. 2017). Limited studies have been found on fault-tolerant
69 or self-correcting controls for building HVAC systems. The purpose of fault-tolerant controls is to
70 help good system operation despite the presence of faults (Zhang and Jiang, 2008). Wang et al.
71 (2002) developed a supervisory control strategy that adapts to the presence of outdoor air flow

72 rate sensor error. Hao et al. (2005) applied principal component analysis to develop fault tolerant
73 HVAC controls. Bengesa et al. (2015) developed a fault-tolerant optimal control schema for a
74 HVAC system integrating FDD and model predictive control. These advanced controls are still in
75 the early research and development stage and are not yet readily deployed in today's BAS.
76 Additionally, these fault-tolerant controls typically do not integrate with or make use of modern
77 FDD technologies, which are becoming increasingly present in commercial buildings.

78 Regarding the topic of automated fault correction, Fernandez et al. (Fernandez et al., 2009a;
79 Fernandez, et al., 2009b) and Brambley et al. (Brambley et al., 2011) developed both passive
80 and proactive fault auto-correction algorithms for an air-handler unit (AHU) and a variable-air-
81 volume (VAV) box. The developed algorithms correct the following faults: temperature and
82 humidity sensor bias, incorrect damper operation, control hunting, and manual overrides. A subset
83 of these algorithms (sensor bias and minimum outdoor air damper position) were tested in a
84 laboratory experiment. They have not been validated in real buildings or integrated into existing
85 BAS and commercial FDD products.

86 Lin et al. (2020a) complemented and extended the work of Fernandez and Brambley (Fernandez
87 et al., 2009a; Fernandez, et al., 2009b, Brambley et al., 2011), by developing fault auto-correction
88 algorithms designed to be integrated with commercial FDD tools. The new auto-correction
89 algorithms afford the FDD technology a certain degree of control capability, as the autonomous
90 correction of faults are enabled by opening 2-way interfaces between the BAS and the FDD tool.
91 These algorithms target incorrectly programmed schedules, override not released, sensor bias,
92 control hunting, rogue zone, and suboptimal setpoints in HVAC systems. All the algorithms
93 developed in the study follow a general auto-correction process, with different control variables
94 overwritten in the BAS, and different ways to determine the correct or improved value of these
95 variables (Lin et al. 2020a). In this process, after the FDD algorithm generates a fault flag for a
96 specific fault, the auto-correction algorithm is initiated to correct this fault. Having a variable in the
97 BAS that is accessible by the FDD tool is the key element in the process. The developed auto-
98 correction algorithms were integrated into commercial FDD tools and some preliminary integration
99 challenges and solutions were documented in Lin et al. (2020b). Among the auto-correction
100 algorithms, three algorithms (rogue zone, improve AHU supply air temperature setpoint reset, and
101 improve AHU static pressure setpoint reset) were deployed in a single commercial FDD software
102 and tested in two office buildings. These preliminary field testing results are presented in Lin et al.
103 (2020a) and Lin et al. (2021). The enhanced FDD tool with these three auto-correction algorithms
104 was able to correct faults successfully. While these preliminary results are encouraging, they fall
105 short of demonstrating all the algorithms and they are limited to a single software platform.

106 This article presents and discusses the final results of the project introduced in Lin et al. (2020a),
107 focusing on their implementation in two commercial FDD tools, and extensive field testing
108 performed in four buildings from late 2019 to 2021. The research team is composed of
109 researchers, FDD implementation partners and facility managers testing the new FDD features in
110 their buildings. This paper presents modifications to the FDD tools and the BAS that were required
111 to enable the execution of different types of auto-correction algorithms in real buildings. It also

112 presents the field testing procedure and results of seven auto-correction algorithms across four
113 buildings and three different BASs. This study aims to answer the following three questions:

- 114 1. Can auto-correction algorithms be successfully implemented in modern FDD tools and
115 field tested in real buildings?
- 116 2. Are the enhanced FDD solutions able to correct faults in real buildings without adverse
117 operational effects?
- 118 3. What are the benefits, adoption drivers, and scalability challenges of fault auto-correction
119 capability?

120 The rest of the paper is organized as follows: section 2 describes the method used for
121 implementing, deploying and testing the algorithms, section 3, 4, and 5 present the FDD tool
122 implementation results, the field tests results, and benefits and challenges of fault auto-correction,
123 respectively. Section 6 summarizes the conclusions.

124 **2. Method**

125 In this study, a set of seven fault-correction algorithms for HVAC systems were tested in real
126 buildings by two FDD partners (Table 1). The development of these routines is described in detail
127 in Lin et al (2020a). The variables corrected by the algorithms span schedules, setpoints, sensor
128 readings, commands, heating/cooling requests, and proportional, integral, derivative (PID)
129 parameters. The algorithms were created based on a detailed literature review and domain
130 expertise of the research and implementation team.

131 The routines can be broadly divided into three types (Table 1):

- 132 • One-time correction. After detecting and identifying the fault, the algorithm corrects it
133 automatically. The correction is not re-triggered until the fault is detected again (Section
134 3.2.1). Algorithms #1-3 are one-time corrections of faults.
- 135 • Active testing + one-time correction. Similar to the one above, with the addition of one or
136 more active tests performed before the variable is overridden in the BAS. The active tests
137 perturb the system in specific operating conditions to determine the best values of
138 parameters to be corrected (Section 3.2.2). Algorithm #4: Control Hunting is in this
139 category.
- 140 • Continuous optimization. After identifying the opportunity, these routines act continuously
141 to optimize system operation, behaving similarly to a “continuous” control algorithm. The
142 overwriting of BAS variables happens with higher frequency than the other two types of
143 routines and human intervention is not required to authorize each BAS variable update,
144 although the algorithm may require initial operator’s approval (Section 3.2.3). Algorithms
145 #5-7 belong to this type.

146 The selected auto-correction routines were implemented into two FDD products (Section 2.1).
147 Then field tests were performed in four commercial buildings (Section 2.2) following the same
148 testing procedure (Section 2.3).

149

Table 1: Summary of the seven auto-correction algorithms implemented and tested in this study

#	Fault/Opportunity Name	Fault/Opportunity Description	Type of Correction	Variables Corrected
1	HVAC schedules are incorrectly programmed	HVAC equipment doesn't turn on/off according to intended schedule due to error in control programming	One-time correction	Schedule
2	Override not released	Operator unintentionally neglects to release what was intended to be a short-term override of setpoints or other control commands (e.g. fan VFD speed, valve control command).	One-time correction	Override property of setpoint or command
3	Improve zone temperature setpoint setback	The zone temperature cooling setpoint is lower than needed or the heating setpoint is higher than needed while the space is scheduled occupied or unoccupied.	One-time correction	Zone temperature setpoint
4	Control hunting	The actuator operates under oscillation due to improper PID parameter setting	Active testing + one-time correction	PID parameters
5	Rogue zone	A zone continuously sends cooling/heating requests, due to zone-level equipment problems like a leaky reheat valve, a dysfunctional supply air damper, or unachievable zone temperature setpoints.	Continuous Optimization	Number of ignored requests from zones
6	Improve AHU static pressure setpoint reset	Non optimized AHU static air pressure setpoint	Continuous Optimization	Supply static pressure setpoint
7	Improve AHU supply air temperature setpoint reset	Non optimized AHU supply air temperature setpoint	Continuous Optimization	Supply air temperature setpoint

151 2.1 Implementation of auto-correction routines into FDD tools

152 The research team first developed the high-level algorithms in the form of flow charts (Lin et al
153 2020a). Later the FDD partners selected a subset of them to implement and deploy them based
154 on desired new features for their platforms and the interest of their clients (Table 2). Partner 1 is
155 an end-user with the staff and internal capability to customize the platform for their needs.
156 Therefore, the routines developed by Partner 1 are site-specific customizations of the standard
157 vendor platform. The FDD tool used by partner 1 is located on the premises and has direct access
158 to the BAS network. This allowed Partner 1 to more easily implement continuous optimization
159 routines and more complex BAS modifications. Partner 2, instead, is a FDD provider with a
160 centralized, cloud-based platform without direct access to the BAS network. The algorithms were
161 developed in a "sandbox" environment where initial testing of the auto-correction functionality took
162 place. Once functionally tested and validated, these new platform features were incorporated into
163 the "production" version of the software for deployment to the test buildings, making this capability
164 also available to other customers while focusing on easily scalable algorithms.

165 Implementation activities included: (1) Modifying the FDD tool and the BAS to enable write
166 capability into the BAS and to set up user interfaces for building operators. These changes are
167 typically software modifications (creation of new points, interface programming, BAS logic

168 changes), but can also include hardware changes or additions (e.g., a new auto-correction device.)
 169 (2) Coding the algorithms in the analytics engine of the FDD tool (3) Commissioning the algorithms,
 170 including a review of the auto-correction algorithm outputs. The results of this implementation step
 171 are described in Section 3.

172 *2.2 Testing sites and equipment*

173 FDD Partner 1 deployed the algorithms on two buildings in the same campus, while Partner 2
 174 deployed them in two separate locations. The testing equipment includes AHUs, variable-air-
 175 volume boxes (VAV), fan coils (FC) and a heat recovery ventilation (HRV) unit for a total of 225
 176 distinct pieces of equipment. The routines were also integrated with three different BAS vendor’s
 177 platforms: Automated Logic Controls¹ (ALC), Johnson Controls Inc.² (JCI), and Delta Controls³
 178 (DC). Table 2 summarizes sites, equipment, BAS and tested algorithms. The tests were
 179 performed between the end of 2019 and the beginning of 2021.

180 Table 2: Summary of the field testing sites and equipment

FDD Partner	Site	Location	Equipment Tested	Algorithm Tested	BAS
Partner 1	Site A	Berkeley, CA, US	2 AHU, 48 VAV	4. Control hunting 5. Rogue zone	ALC
	Site B	Berkeley, CA, US	2 AHU, 163 VAV	5. Rogue zone, 6. Improve AHU supply air static pressure setpoint reset 7. Improve AHU supply air temp. setpoint reset	JCI
Partner 2	Site C	Vancouver B.C. Canada	3 FC and 1 HRV	1. HVAC schedules are incorrectly programmed 2. Override not released 3. Improve zone temp. setpoint setback	DC
	Site D	Atlanta, GA	1 AHU and 6 VAVs		DC

181 *2.3 Testing procedure*

182 After implementing the algorithms and deploying them into the buildings, their operation was
 183 tested using the following procedure. The procedure aimed at assessing the ability of an FDD tool
 184 to automatically correct a fault, and therefore allowed the “correction” capability of the FDD tool
 185 to be decoupled from its ability to perform detection and diagnostics. In this way, potentially
 186 confounding factors can be ignored, associated with false negative/positive detection or
 187 incorrect/missed diagnosis.

188 For each fault and automated fault correction procedure, each implementation partner did:

- 189 1. Verify the ability to override all setpoints/parameters to be tested.

¹ <https://www.automatedlogic.com/en/>

² <https://www.johnsoncontrols.com/>

³ <https://deltacontrols.com/>

- 190 2. Use a naturally occurring fault or impose the fault in “clean” (fault-free) equipment or
191 “assume” the fault is present if physical presence of the fault is not necessary to validate
192 the behavior of the corrective action.
193 3. Observe and document the FDD tool output, i.e., its detection and diagnosis results (not
194 applicable for assumed faults).
195 4. Execute the FDD-embedded correction routine.
196 5. Observe and document the effect of the automated fault correction.
197 The results of the tests are described in Section 4.

198 *2.4 Interviews with partners, facility managers and industry advisors*

199 At the end of the project, the research team conducted a series of interviews with seven different
200 FDD providers and two facility managers. The researchers asked questions about perceived
201 benefits of auto-correction, as well as market barriers and potential drivers of adoption of this
202 technology. The interviews were transcribed and their content is summarized in Section 5, to
203 support answering the third research question.

204 **3. FDD tool and BAS modifications for auto-correction**

205 Partner 1 implemented one *active testing + one-time correction algorithm* and *three continuous*
206 *optimization algorithms*. Partner 2, instead, implemented three *one-time correction algorithms*.
207 These are listed in Table 2.

208 *3.1 FDD-BAS infrastructure update*

209 The current state-of-art FDD systems typically use one-way communication with the BAS, reading
210 operational data, running analytics, and flagging faults on the software interface (Lin et al., 2020b).
211 The first step in the software development for both partners consisted of enabling secure 2-way
212 communication between the FDD tool and the BAS.

213 *3.1.1 Partner 1 implementation*

214 Due to cybersecurity requirements of the site for Partner 1, the FDD software is hosted on a server
215 within the site’s firewall protected internal networks. The server collects data directly from
216 BACnet/IP networks, and by existing on those networks it is also capable of issuing BACnet
217 commands⁴. For this reason, access to the server is restricted to administrators. Data from the
218 FDD software is replicated to a separate server for end-user access to visualization and reporting
219 tools, accessible remotely. All FDD auto-correction routines, applications, and point mappings
220 reside on the internal server. The architecture of the FDD tool and the BAS developed by partner
221 1 is presented in Figure 1a. The blue line shows the original infrastructure and the red line shows
222 the upgrade. Before the start of the project the FDD tool already included a BACnet module, which
223 implemented the BACnet communication protocol (ASHRAE, 2021), to extract data from the BAS.
224 This BACnet module already enabled two-way communication, but each writable setpoint or

⁴ For more information about known cybersecurity vulnerabilities related to the BACnet protocol, the reader should consult Holmberg and Evans, (2003) and Peacock et al. (2018).

225 command needed to be further configured in the FDD tool to enable writing operations and to
 226 define BACnet priority levels⁵. Several modifications also needed to be made in the BAS to make
 227 sure that the auto-corrected controls could operate even if the connection with the FDD tool was
 228 lost, described in Section 3.2.3.

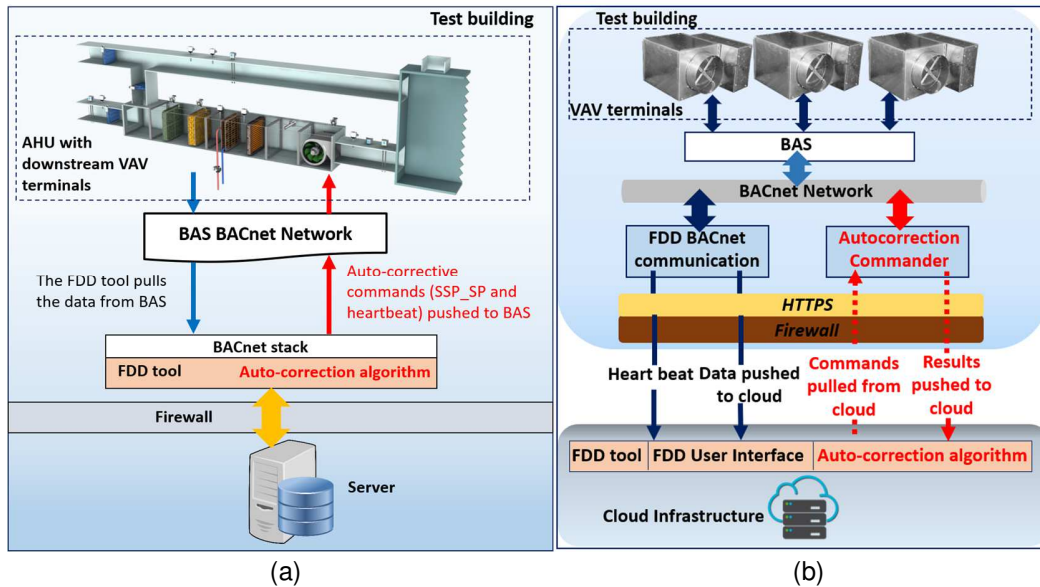


Figure 1: New FDD-BAS architecture created to support auto-correction (a: Partner 1, b: Partner 2)

3.1.2 Partner 2 implementation

233 The architecture of the new system is shown in Figure 1b. In traditional deployments, the most
 234 common FDD integration pathway for Partner 2 involves the installation of a local device within
 235 the BAS infrastructure. Once online, this device is tasked with systematically polling the
 236 networked devices to retrieve configuration and operational data, continuously delivering these
 237 data sets to the cloud servers for storage and analysis. The data-collection device is securely
 238 connected to the cloud platforms by limiting its interaction with a specified IP address and only
 239 initiating outbound messages from the site to the cloud. The existing FDD algorithms of the
 240 standard platform are run on the cloud and accessed via a web interface from any computer with
 241 the proper credentials.

242 To enable two-way communication to the FDD platform, Partner 2 opted for adding a new device
 243 – Autocorrection Commander, which manages the execution of the auto-correction algorithms.
 244 After BAS data is collected and pushed to the cloud platform, the new auto-correction algorithms
 245 are run in the cloud and the correction commands are prepared for execution. The new device
 246 periodically pulls these commands from the cloud and executes them on the BAS network. When
 247 the BAS receives the correction command, correction actions are implemented. The correction
 248 results are collected via the new device and pushed back to the cloud FDD platform. Compared
 249 to Partner 1, this implementation requires more attention to be paid to synchronization between
 250 cloud intelligence and local execution, because loss of connectivity is more likely to occur. Partner

⁵ BACnet uses priority levels as a mechanism to assign priority to specific entities to prevent conflict between control actions.

251 2 implemented two features in the new device to avoid synchronization issues. The first feature
 252 is “value validation” which means the device will validate the value that has been collected (in
 253 case this value is changed after the FDD results are delivered) before it attempts to auto-correct
 254 it. If the value is as expected, the auto-correction would proceed, if it is different, it would deny
 255 auto-correction and insert an explanation of this in the activity log. The second feature is
 256 "command expiration" in case loss of connectivity delays the ability for the auto-correction device
 257 to communicate to the cloud and get the latest correction commands from the queue.

258 *3.2 Software development for algorithms, BAS integration and UI*

259 In addition to modifying the platform to enable two-way communication, each partner translated
 260 the research-grade algorithms generated by the research team into platform-specific auto-
 261 correction algorithms. This was accomplished by using the native scripting language of each
 262 platform. Other software modifications were required in both the FDD tool and the BAS, for
 263 example to create and integrate new points, to generate user interfaces, or to modify the BAS
 264 logic. Sections 3.2.1-3.2.3 describe the translation and Table 3 summarizes other software
 265 modifications.

266 Table 3: Software modifications in the FDD tool and the BAS in addition to the translation of the
 267 algorithms

#	Fault/Opportun-ity	FDD tool modification	BAS modification
1	HVAC schedules are incorrectly programmed	<ul style="list-style-type: none"> - Create new FDD writable point⁶ (schedule) - Modify the user interface - Create an action log 	- No algorithm-specific modification
2	Override not released	<ul style="list-style-type: none"> - Create new writable point (override) - Modify the user interface - Create an action log 	- No algorithm-specific modification
3	Improve zone temperature setpoint setback	<ul style="list-style-type: none"> - Create new writable point (setpoint) - Modify the user interface - Create an action log 	- No algorithm-specific modification
4	Control hunting	<ul style="list-style-type: none"> - Integrate PID parameters as new points - Create test management application - Create new database tables for PID loop info - Create test log 	- Expose PID parameters to BACnet (when not available by default)
5	Rogue zone	<ul style="list-style-type: none"> - Create 1 new ignored requests⁷ point for each zone (if desired for logging) - If not already present, add FDD rules related to rogue zone detection (e.g. leaky reheat valve) - Create ignore calculation application - Integrate new writable AHU 'ignore' points 	<ul style="list-style-type: none"> - Add BACnet-exposed 'ignore' inputs to existing AHU control logic. - If logic doesn't already exist, calculate new effective heating and cooling requests, using provided 'ignore' inputs

⁶ Note: BAS and FDD tools typically store time-series data into a database sometimes called “historian”. By “point” we mean new variables linked to these time-series data.

⁷ Requests and ignored requests (also called 'ignore') are defined in section 3.2.3

6	Improve AHU static pressure setpoint reset	<ul style="list-style-type: none"> - Create FDD writable point (setpoint) - Integrate required zone points if not already available (e.g. cooling PID output) 	<ul style="list-style-type: none"> - Create new BAS point (setpoint) - Create new point for FDD heartbeat - Modify control sequence to use the new static pressure/supply air temp. setpoint when heartbeat is present - Modify BAS graphics to provide operator transparency and control.
7	Improve AHU supply air temp. setpoint reset		

268

269 3.2.1 One-time Correction

270 Partner 2 developed platform-specific algorithms #1: *HVAC schedules are incorrectly*
 271 *programmed*; #2: *Override not released* and #3: *Improve zone temperature setpoint setback*
 272 described in Table 1. The underlying FDD tool already saves several parameters describing the
 273 intended operation of the buildings, including schedules, control modes, and setpoints. These
 274 “recommended” parameters are actively determined from the operation history or selected by the
 275 facility managers. In the FDD tool, these parameters are continuously compared with current
 276 schedules and operation, and when the operation deviates from the recommended values, the
 277 facility managers are notified. In the standard implementation of the software, the user is required
 278 to use the BAS interface to revert the parameters back to the saved value or change this saved
 279 value at the FDD if the BAS value is deemed the more appropriate value. With auto-correction,
 280 the software was modified to allow updates of these parameters from the FDD tool. In order to
 281 accomplish that, the user interface was modified to show the recommended value and a log of
 282 the previous auto-correction actions (a and b in Figure 2 respectively). The end-users have the
 283 option to either approve the auto-correction action that reverts the values back to the
 284 recommended value or confirm the latest value is correct and update it to be the new
 285 recommended value. This additional evaluation step was adopted to earn facility staff’s trust. The
 286 software also includes an option to enable automatic correction of faults, bypassing the approval,
 287 once the building manager has gained trust in the system. These two alternative paths are also
 288 represented in Figure 9 (i.e., correction evaluation and auto-approval). When executing the auto-
 289 correction actions for all three algorithms, the FDD tool changes the value of the related BACnet
 290 variable (i.e., *Weekly_Schedule* for #1, *Out_Of_Service* property for #2 and *Present_Value* for #3
 291 respectively) to recommended values. On the BAS side, no change was necessary for each
 292 algorithm, aside from opening two-way communication between the FDD and BAS platforms.

Object: HQ (AC11) Min Damper Position

PROPERTY				
PRIMARY DATA	Latest Value		Recommended Value	
Value	1	34	1	30
Units				

Action History

Automatic System Optimization changed status to Closed & commented "ASO: 200 Success Value now[30] was [33.599998]"
February 10, 2020 03:00 PM

Automatic System Optimization changed status to In-Progress & commented "ASO: Queued"
February 10, 2020 03:00 PM

293

317 (b) Management of active tests

318 As the start of the auto-correction process, this module initiates the active test following the
319 Lambda open-loop tuning rule (Pruna et al, 2017) which determines the improved PID parameters
320 based on the open-loop reaction of the process variable (e.g. temperature) to a change in the
321 control variable (e.g. valve control command).

- 322 • A persistent task runs every hour to review all the PID loop records that have auto-testing
323 enabled. If a PID loop requires a test (number of successful tests < target) and a test
324 wasn't recently completed (in the last 24 hours), the software queues up a new test by
325 creating a PID test record, and a new temporary, dedicated task.
- 326 • Temporary, dedicated tasks run every 5 minutes to perform the following steps
327 sequentially:
 - 328 • Load its dedicated PID test record,
 - 329 • Synchronize time-series data to have the most recent data available for the
330 process and control variables,
 - 331 • Check if testing conditions are met (e.g. airflow is detected for a reheat valve test),
 - 332 • If testing conditions are not met, the current test fails.
 - 333 • If testing conditions are met, monitor previous changes in the control variable and
334 the corresponding reaction of the process variable to determine what action, if any,
335 needs to be taken:
 - 336 ○ Override control variable to achieve stable state (start of test)
 - 337 ○ Override control variable to perform step change
 - 338 ○ Release override (end of test)

339 When enough data has been collected, module (c) is called.

340 (c) Calculation of improved PID parameters

341 This module is invoked during testing, if enough data has been collected after the execution of
342 module (b). In this module, the improved PID parameters are calculated using the collected data
343 of the control variable, the process variable, and the time between the step change in control
344 variable and the response from the process variable. If the module then successfully returns
345 improved PID parameters, the test is considered to be complete and successful.

346 (d) Database and interface to access test results

347 This module stores and views the results from modules (b) and (c). Information about each test
348 is recorded in a database, including start and end timestamps, whether the active test is
349 successful or failed, and the PID parameters' results from the successful tests. A user interface
350 is also created that allows for viewing these results.

351 3.2.3 Continuous Optimization

352 Partner 1 implemented three algorithms #5, #6, and # 7 aimed at improving AHU operation using
353 supply air temperature and static pressure resets, enhanced by an evaluation of rogue zones.
354 These strategies are ranked in the top ten efficiency measures implemented by organizations
355 using FDD technology based on FDD analytics results. (Kramer et al., 2020). The auto-correction
356 algorithms for this opportunity are closely related to ASHRAE High-Performance Sequences of
357 Operation Guideline 36 (ASHRAE, 2018), but deployed via the FDD tool instead of the BAS (Lin

358 et al., 2020a). These algorithms determine the values of setpoints depending on the number of
 359 cooling “requests” generated by downstream zones that are served by the same AHU and write
 360 the improved setpoints into the BAS every five minutes. Details about their implementation are
 361 described in Lin et al. 2020a and Lin et al. 2020b. Table 3 summarizes the modifications
 362 necessary to their operation. These include creation of new points and new logic in the FDD tool
 363 and modification of interfaces and control sequences in the BAS. Heartbeat signals were also
 364 added in the FDD tool and sent to the BAS to constantly monitor connectivity between the two
 365 systems. If the BAS lost connection with the FDD tool it would revert back to the output of the old
 366 control sequence.

367 **4. Field testing results**

368 After implementing the routines and debugging them, each partner conducted formal field tests in
 369 real buildings, following the procedure highlighted in Section 2.3. The implementation partners
 370 successfully tested the algorithms, without adverse consequences, in at least one building and
 371 HVAC system. In two cases (algorithms #1 and #4) the code had to be modified to address
 372 problems identified during the field test. In seven cases, the faults were artificially imposed on the
 373 system, in order to test the procedure and in nine cases, the faults were successfully detected by
 374 the FDD tool. In two cases the FDD tool did not have the detection algorithms and the faults or
 375 opportunities were practically determined by the facility staff (N/A). The results of these field
 376 studies are summarized in Table 4 and described by type of algorithm in Section 4.1-4.3.

377 Table 4: Summary of test results of field testing in four buildings

#	Algorithm Tested	Site	Equipment	Artificially imposed	Fault detected	Auto-correction without adverse impact
1	HVAC schedules are incorrectly programmed	Site C	1 FC	Y	Y	Y
		Site D	1 AHU	Y	Y	Y
2	Override not released	Site C	3 FCs and 1 HRV	Y	Y	Y
		Site D	3 VAVs	Y	Y	Y
3	Improve zone temp. setpoint setback	Site C	3 FCs	Y	Y	Y
		Site D	6 VAVs	Y	Y	Y
4	Control hunting	Site A	1 VAV	Y	Y	Y
5	Rogue zone	Site A	2 AHU, 48 VAV	N	Y	Y
		Site B	2 AHU, 163 VAV	N	Y	Y
6	Improve AHU static pressure setpoint reset	Site B	2 AHU, 163 VAV	N	N/A	Y
7	Improve AHU supply air temp. setpoint reset	Site B	2 AHU, 163 VAV	N	N/A	Y

378 *4.1 Testing results of one-time correction algorithms*

379 Partner 2 tested algorithm #1 on two pieces of equipment in two sites, algorithm #2 on seven
 380 pieces of equipment and algorithm #3 on nine pieces of equipment across two sites (Table 4). All
 381 the faults were artificially imposed on the equipment and then the facility manager executed
 382 corrections after the faults were flagged in the FDD tool.

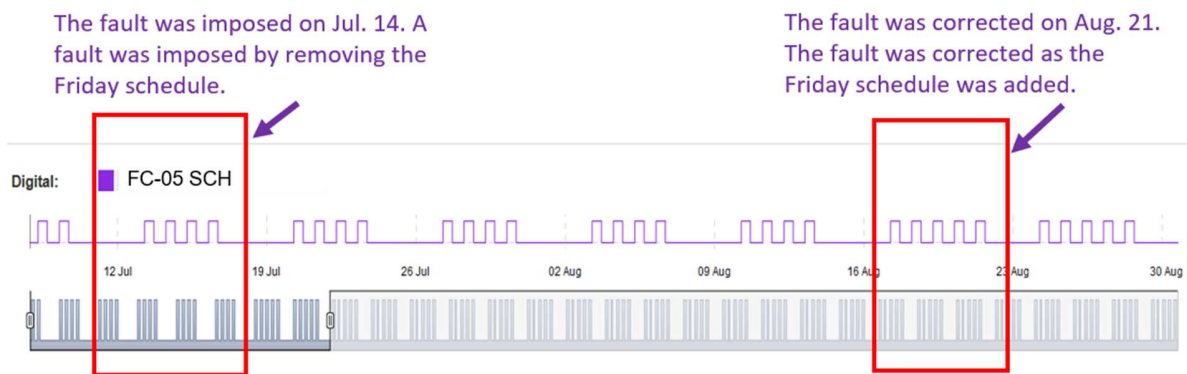
383 For algorithm #1: *HVAC schedules are incorrectly programmed*, two cases failed and two were
 384 successful in the two testing sites. The two early tests failed due to inconsistencies in the

385 implementation of the schedule object in the BAS. Two additional tests were successful after
 386 updates in the auto-correction code. An example of the successful results is presented in Figure
 387 4 (a). The equipment schedule was modified by purposely deleting Friday's schedule set to 6:30
 388 AM - 5:00 PM on the BAS object property 'TEMP_SCH'. The fault was correctly identified by the
 389 FDD tool on July 15. Figure 4 (a) shows the Friday schedule disappeared after the fault was
 390 implemented and reappeared on August 21 after the correction action was authorized by the user.
 391 Between the start of the test on July 1 and the successful correction on August 21, the first test
 392 failed due to an integration issue between the BAS and the FDD platform. This required changes
 393 in the FDD software that was resolved the second week of August. Following this software update,
 394 the correction operated as expected.

395 For algorithm #2: *Override not released*, the correction succeeded in all test cases at the two test
 396 sites. The results of an example test case "The zone temperature setpoint mode of a FC was
 397 overridden from auto to manual" is presented in Figure 4 (b). The value of "zone temperature
 398 setpoint mode" was changed from 0 (auto) to 1 (manual) when the fault was imposed on August
 399 20. Auto-correction was executed at 18:00 on August 21 after the fault was detected, and
 400 successfully changed back the value from 1 (manual) to 0 (auto). In the other test cases, the
 401 mode of other setpoints (i.e., maximum, minimum, or actual space temperature setpoints, the
 402 night-heating setback enable temperature setpoint, and the CO₂ differential setpoint) were
 403 overridden from auto to manual, and the algorithms also successfully converted them back to
 404 auto.

405 For algorithm #3: *Improve zone temperature setpoint setback*, the values of zone temperature
 406 setpoints were changed to impose faults. All the faults were successfully corrected without
 407 adverse impact. For example, during the test of a FC in Site C, the actual zone temperature
 408 setpoint was changed from 21 °C to 19 °C at 5:00 PM August 20 to impose the fault. Figure 4 (c)
 409 shows the zone temperature and setpoint between August 20 and August 22. After the fault was
 410 imposed at 5:00 PM, the zone temperature setpoint was decreased from 21°C to 19°C. As a result,
 411 the zone temperature dropped until it reached the new setpoint of 19°C two hours later. The next
 412 day (i.e., August 21), the zone temperature tracked the wrong zone temperature setpoint until the
 413 correction action was executed at 11:00 AM August 21. Consequently, zone temperature reached
 414 the corrected setpoint value 21°C.

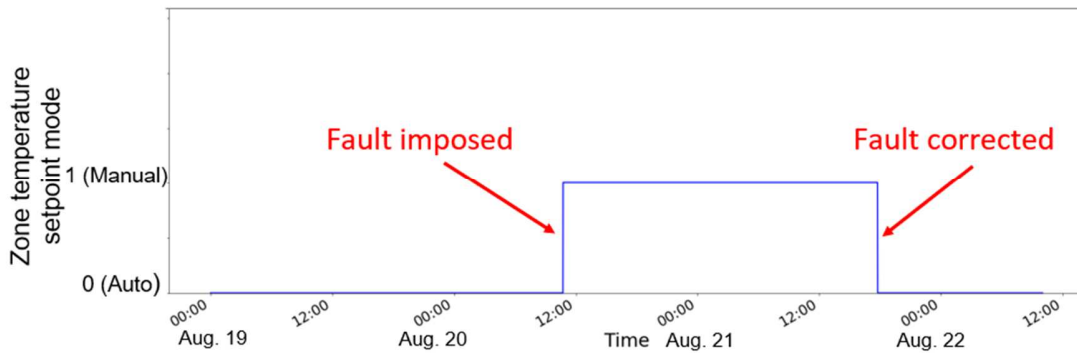
415



416

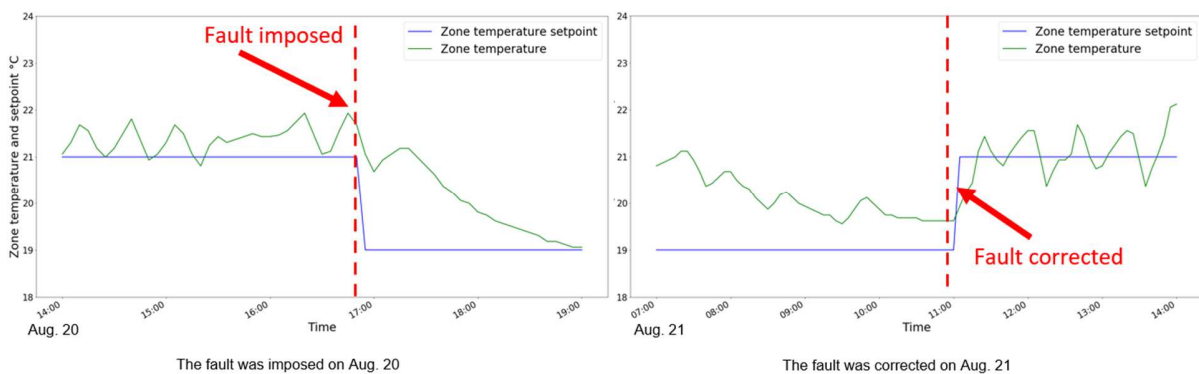
417

(a)



418
419

(b)



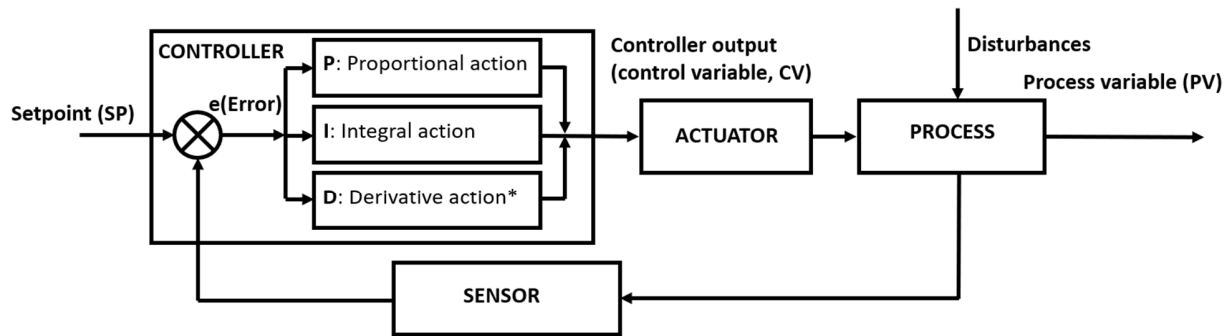
420
421

(c)

422 Figure 4: Example auto-correction test results (a): #1 HVAC schedules are incorrectly programmed, (b):
423 #2 Override not released, and (c): #3 Improve zone temperature setpoint setback (left: the fault was
424 imposed on Aug. 20; right: the fault was corrected on Aug. 21)

425 **4.2 Testing results of active testing + one-time correction algorithm**

426 Partner 1 tested algorithm #4 *Control hunting* on a VAV box discharge air temperature control. In
427 this control loop, the PID controller compares the setpoint to the discharge air temperature
428 (process variable) to obtain the error, then the reheat valve command (control variable) is
429 determined based on the error and PID parameters. The reheat valve command inputs to the
430 actuators to generate actual control actions so that the discharge air temperature reaches the
431 setpoint (Figure 5)



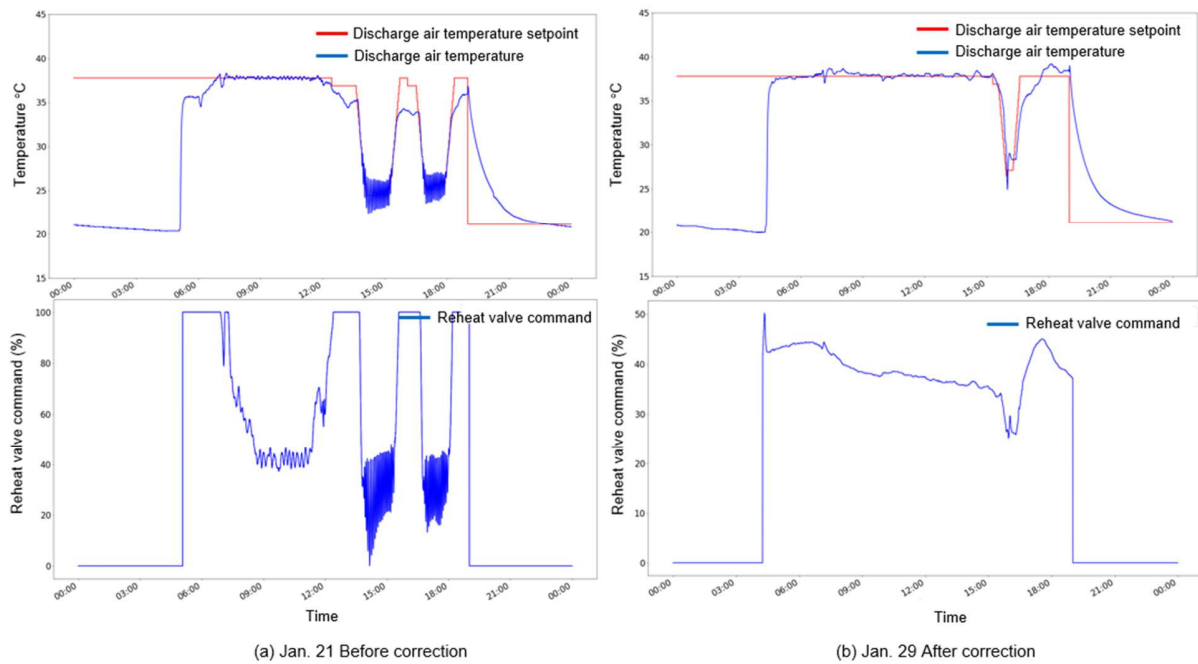
*D=0 as a PI controller

Figure 5: Control loop, control variable and process variable for test of algorithm #4

432
433

434 The successful test was conducted in January 2021. The behavior of the control and process
 435 variables on January 26th, 2021, before auto-correction, is displayed in Figure 6a. The top panel
 436 shows the Discharge Air Temperature (process variable, in blue) and the Discharge Air
 437 Temperature Setpoint (in red). The temperature oscillated more than 15 times per hour between
 438 2-3pm and between 5-6pm. The oscillations were caused by the control variable, the Reheat
 439 Valve Command, displayed in the bottom panel (in blue). This hunting behavior was caused by
 440 improper PID parameters. Figure 6b shows trends from the same points on January 29th, 2021,
 441 after the auto-correction routine was executed. The oscillations of control and process variables
 442 (i.e., hunting) disappeared after the update of the parameters, and a hunting fault was no longer
 443 detected by the FDD tool.

444 To calculate the improved PID parameters for correction, the implementation team performed an
 445 active perturbation test, as described in Section 3.2.2. Figure 7 displays trends and the active test
 446 results. The results include proposed PID parameters - proportional and integral gains (K_p and
 447 K_i) determined from the test. To perform the open-loop step change test, the FDD tool increased
 448 the control variable (Reheat Valve Command of the VAV) from 45% to 65%. As a result, the
 449 Discharge Air Temperature increased from 38.4 °C to 40 °C. The improved PID parameters,
 450 calculated from Lambda open-loop tuning rules (Pruna et al, 2017), were proportional gain(K_p) =
 451 0.5 and integral gain (K_i) = 0.1. Derivative gain is zero for PI controls, frequently used in HVAC
 452 control systems.



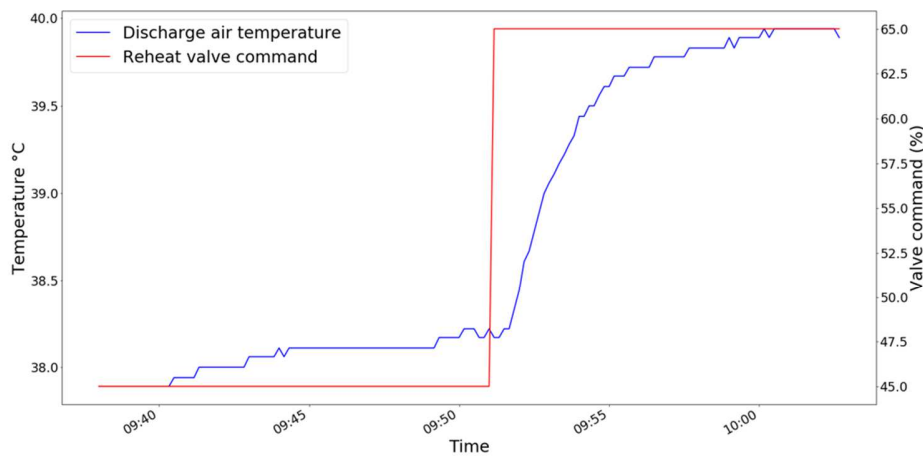
453

454

455

456

Figure 6: Comparison of behavior of the Discharge Air Temperature (process variable) and Reheat Valve Command (control variable) in Site A before (Jan 21th 2021, a) and after (Jan 29th 2021, b) the update of the PID parameters.



ALC K _p : proportional gain	ALC K _i : integral gain	ALC integral
0.5	0.1	1 min

457

458

459

460

Figure 7: Process variable (Discharge Air Temperature), control variable (Reheat Valve Command) and derivative of the process variable for the test of algorithm #4. The bottom table shows the results of the calculation of improved PID parameters in the FDD interface (ALC K_p and K_i)

461 *4.3 Testing results of continuous optimization correction algorithms*

462

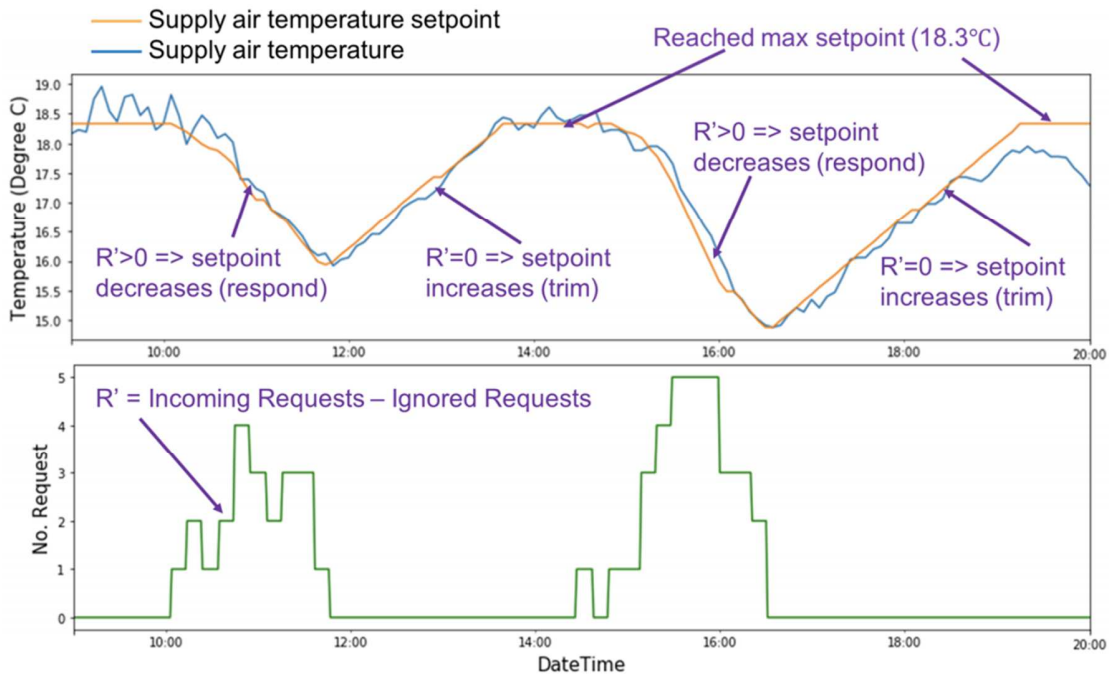
463

Partner 1 tested algorithm #5 in two buildings and #6 and #7 in a single building. For each building, the routines were implemented on two large AHUs serving tens to hundreds of VAV boxes.

464 Detailed results of the preliminary tests of algorithms #5, #6, #7 are presented in Lin et al., 2020a,
 465 and Lin et al., 2020b. The new control strategies have been permanently adopted by the site
 466 beyond the testing requirements of the project and have now been running for over a year. The
 467 new control sequence did not cause any occupant complaints, and it worked more efficiently than
 468 the previous ones, although precise savings estimates were beyond the scope of the test. Figure
 469 8 shows results when the FDD tool successfully changed the supply air temperature setpoint of
 470 one of two test AHUs based on the algorithms described in Section 3.2.3. The bottom of Figure 8
 471 describes the number of calculated requests R' , defined as:

$$\text{Calculated (Heating or Cooling) Requests (R')} = \text{Incoming Requests (R)} - \text{Ignored Requests (I)} \quad (2)$$

472 When the number of R' became larger than zero starting at 10:05 a.m., the algorithm slowly
 473 reduced the SAT setpoint by 0.06 °C for each request every five minutes. Starting at 11:50 a.m.,
 474 the requests remained at zero and the routine slowly increased the supply air temperature
 475 setpoint by 0.12 °C every five minutes until it reached a max value (SAT_{max}=18.3 °C). The
 476 setpoint remained at SAT_{max} until R' was positive again at 14:50 p.m. Then, the supply air
 477 temperature setpoint again slowly decreased when R' was positive and slowly increased when R'
 478 became zero. The strategy saved energy compared to the legacy control algorithm as illustrated
 479 in Lin et al., 2020a.



480
 481 Figure 8: The SAT setpoint of an AHU after the execution of the auto-correction algorithm (Lin et al.
 482 2020a)

483 **5. Benefits and challenges of fault auto-correction**

484 *5.1 Technology benefits and market drivers*

485 Commercial FDD platforms help to continually identify operational inefficiencies in building
486 equipment. However, these FDD tools generate recommendations that need to be implemented
487 by service technicians or other staff, resulting in delays, operations and maintenance costs or lost
488 opportunities. All the FDD providers and facility managers interviewed during the project (Section
489 2.4) recognized these shortcomings and agreed that fault auto-correction integrating with
490 commercial FDD technology offerings can close the loop between the passive diagnostics and
491 active control. Several providers highlighted that many buildings with small operations teams may
492 struggle to respond to FDD fault reports in a timely manner. The ability to auto-correct faults, even
493 if it is only a subset of the total faults list identified by an FDD tool, can make a significant difference
494 in the realized savings. One interviewee asserted that, for many organizations, auto-correction
495 will be the primary way they can scale their ability to act on FDD findings.

496 In particular, the interviewees identified several benefits of this technology:

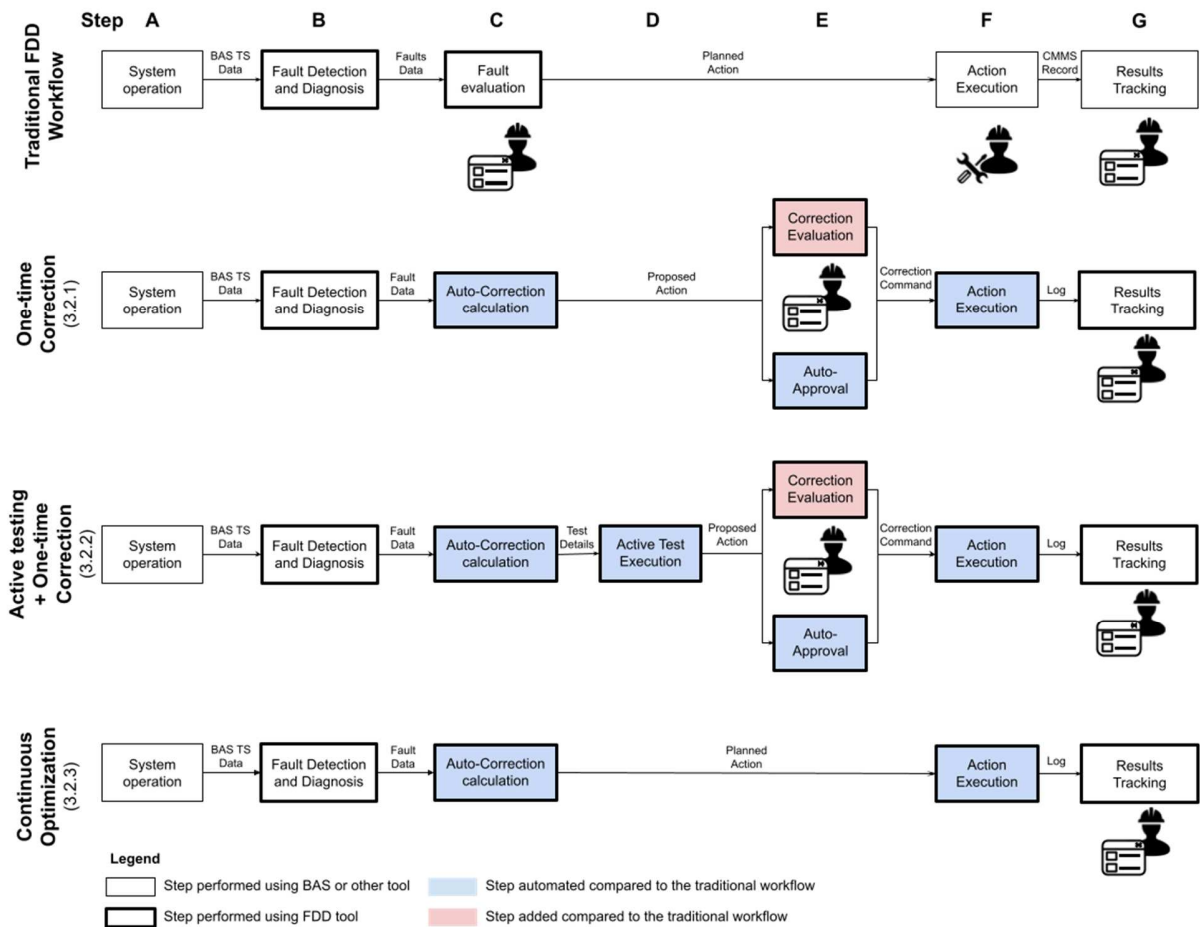
- 497 1. reducing the extent to which savings are dependent upon human intervention
- 498 2. scaling building operators' ability to act on FDD findings (especially for facilities with small
499 operations teams),
- 500 3. tracking the changes executed on the BAS
- 501 4. applying consistent fixes for a subset of fault conditions,
- 502 5. saving a significant amount of energy from the routines related to optimal controls

503 To better understand how the different algorithms tested in this paper enable these benefits, the
504 research team abstracted the workflow of each category of algorithm and compared them to the
505 standard FDD workflow (Figure 9). Each box in Figure 9 represents a step in the process and the
506 arrows indicate the data transferred between them. The steps that involve facility staff are
507 indicated by a human icon, while the automated steps have no icon. The colored boxes represent
508 changes compared to the standard workflows. The tasks in blue are automated by the algorithms,
509 while tasks in red add a new step to the traditional process. The second and third group of
510 algorithms show two parallel paths, because different options may fully automate the task or
511 require human confirmation.

512 In the traditional case, the FDD tool identifies the faults using data from the BAS (step B in Figure
513 9). A human (e.g., facility manager) evaluates these faults and plans a set of actions to fix the
514 identified issues (step C). After this phase, other actors fix physical problems in the underlying
515 systems or reprogram the BAS (step F). The resulting actions are typically, but not always,
516 recorded in a system different from the FDD tool (step G), for instance a computerized
517 maintenance management system (CMMS) (Wireman, 1994). The manual steps in the
518 implementation of the corrective actions and the difficulties in tracking their outcomes are often
519 recognized as limitations of current FDD platforms (Granderson et al., 2017).

520 The algorithms proposed in this paper improve over this base workflow through some degree of
521 automation, but they differ in some of the steps. All the algorithms automate the correction of
522 faults (step F: action execution), thus contributing to reducing the dependency of savings from

523 human intervention (benefit #1) and increasing the ability of the facility team to act on FDD findings
 524 (benefit #2). However, they also replace the manual evaluation of faults (step C, in the basic
 525 workflow) with additional steps that depend on the algorithm group. For the *one-time correction*
 526 algorithms, the detection of the fault triggers a proposed action. The user can manually approve
 527 it or decide to approve it automatically (step E). Partner 2 plans to implement options in the
 528 interface in future that allow users to auto-approve certain corrections, after gaining trust in the
 529 system (Step E, in blue). This feature will allow to apply consistent fixes to a subset of fault
 530 conditions (benefit #4). After this decision, the FDD tool generates a command, pushes it to the
 531 BAS (step F) and tracks it in a log (step G).



532

533

534 Figure 9: Traditional FDD workflow and enhanced process with the three fault correction types

535 The workflow for *active testing + one-time correction* algorithms add an additional step to the
 536 previous process, to gather additional information (step D, Figure 9) used to calculate parameters
 537 and recommend them to the user. Auto-correction routines that involve active testing have
 538 promising applications. For instance, automated tuning of PID loops could save operators the
 539 time to perform trial and error tests of parameters in the field. This is useful for control systems
 540 that don't already have such functionality, or for which out-of-the-box results are not satisfactory.

541 The algorithms belonging to the *continuous optimization* category automate both fault evaluation
542 (step C) and execution of the correction (step F). Since the correction takes place continuously,
543 the operator is only involved in initial approval and periodic evaluations of the outcome of the
544 strategy. Based on the interviews and preliminary test results (Lin et al, 2020a), these strategies
545 have the highest potential for energy savings (benefit #5). Further, these routines may be
546 especially cost-effective on sites where the underlying control infrastructure is obsolete and
547 heterogeneous, because they allow the deployment of supervisory control algorithms more with
548 less labor.

549 In addition, all the algorithms log the corrections enacted by the FDD tool, tracking the changes
550 executed on the BAS (benefit #3).

551 The interviewees agreed that the drivers for market adoption of auto-correction features will be
552 similar to those of FDD tools. For example, energy efficiency and conservation goals are likely to
553 be increasingly important in the future. Labor shortages within the operation and management
554 industry, caused by many facility staff approaching retirement age, may also favor solutions that
555 automate parts of the traditional operation workflows. With common experience of electronics and
556 other consumer devices, facility staff may also expect a better user experience with HVAC
557 controls. A new driver may also emerge as building occupancy patterns are more dynamic post-
558 pandemic, whereby auto-correction via the FDD tool can be a good option to implement the
559 occupancy-based supervisory control strategies.

560 An additional driver is that FDD adoption continues to increase, meaning there is a larger market
561 of established FDD users who will be looking for ways to extend their benefits beyond one-way
562 fault detection.

563 *5.2 Scalability challenges*

564 While the benefits of this technology are significant, several challenges have to be addressed in
565 future research to enable scalable deployment of these algorithms. The first common challenge
566 is enabling secure two-way communication between the FDD tool and the BAS, allowing the FDD
567 tool to override the BAS. The required effort for this integration varies depending on the IT/BAS
568 network architecture. During this project this step was successful on two FDD tools and three
569 types of BAS in three office buildings and one university student center, as described in Section
570 3.1. Additional field testing with more FDD tools and BAS types will be conducted in future to
571 prove the generalizability of these solutions.

572 The second common challenge is overcoming cybersecurity and accountability concerns, when
573 systems are controlled by a third party. The building owners and operators interviewed indicated
574 that, similar to other supervisory control software, they are concerned about remote changes to
575 BAS settings, especially for some building segments like military and healthcare. Interviewees
576 also noted that many owners may be reluctant to hand over any portion of their building's control
577 to a third party. The acceptability of this control overwritten will eventually be determined by the
578 balance between risks and benefits perceived by the organizations using them. To mitigate this
579 challenge, interviewees suggested ensuring the correction routines and corresponding control
580 action be transparent, implementing proposed auto-correction routines only after the confirmation
581 from onsite operations staff, enhancing auto-correction interface to build trust and confidence,

582 and starting with owners who are well established with using FDD and looking for additional
583 benefits. Close attention should be paid to clearly communicating to all the facility staff so that
584 they are aware of the changes to the BAS made by the FDD tool.

585 The research team then evaluated each type of algorithm in relationship to the two dimensions of
586 scalability, 1) required effort, 2) generalizability.

587 The *one-time fault correction* algorithms require low effort (coding and integration) and has high
588 generalizability. They were the simplest to implement, because they modify schedules, setpoints,
589 commands and sensor values, most of which are standard BACnet objects. For the same reason,
590 the implementation partners believe these routines will be easy to scale up across multiple
591 buildings using different BAS, given the growing adoption of BACnet in the control industry.
592 However, the field test demonstrated that even when BACnet is used, different versions of the
593 protocol or proprietary/custom objects used by different BAS vendors may require customization
594 of the code. For example, Partner 2 discovered different implementations of schedule objects in
595 different buildings, some starting the week from Monday, while others from Sunday. These
596 discrepancies in data formatting were found between different versions of BACnet devices as well
597 as in different implementations of the BACnet protocol stack. To ensure user acceptability, Partner
598 2 updated the user interfaces of the FDD tool to allow operators to revise the proposed corrections
599 and to automate the process even further. During the tests, users provided positive feedback and
600 seem to accept these algorithms without problems.

601 The *one-time correction + active testing* algorithm requires high effort and has medium-low
602 generalizability. Several interviewees highlighted the benefits of auto-correction of “control
603 hunting”. Many PID loops in the buildings are out of tune, and it would take significant operator
604 time to manually perform trial and error tests of parameters in the field. In spite of great potential,
605 the effort required to develop this type of *active testing* algorithm was significant, since neither the
606 FDD nor the BAS offered tools to manage the periodic tests needed. As described in Sections
607 3.2.2 and 4.2, three modules and a new interface were added in the FDD tool to understand and
608 manage these tests. Timing and scope of the active testing were managed programmatically, by
609 setting allowed testing times and other conditions based on available trends (e.g. zone
610 temperature) in order to ensure that the desired perturbation of the system did not adversely affect
611 occupant safety or comfort. The generalizability of the auto-correction of PID parameters is
612 medium to low. While the single test with Lambda open-loop tuning rule was successful in a reheat
613 valve - discharge air temperature VAV-box control loop in this study, further work has to be done
614 to fully automate this procedure, prove its robustness, and make it applicable to additional types
615 of equipment. Partner 1 reported that the BAS controller tested did not expose PID loop
616 parameters via BACnet and exposing them required significant manual work. Without proper
617 standardization of BACnet objects and properties describing PID parameters, implementing these
618 routines will require customization to interface with different implementations of PID loops.

619 The *continuous optimization* algorithms also require high effort and has medium-low
620 generalizability. The development was time-intensive, because it required the modification of the
621 BAS logic as well as the FDD tool. The BAS logic cannot be accessed via BACnet and its update
622 currently requires dedicated and proprietary tools that depend on the BAS vendor. While recent
623 research has been exploring how to digitize control sequences (Wetter et al, 2022),
624 standardization of such workflows is still underway. Partner 1 successfully implemented these

625 algorithms in the FDD software which is hosted on a server within the site's firewall protected
626 internal networks. To scale up these algorithms in the cloud FDD software, FDD providers should
627 develop methods to ensure the synchronization between the BAS and the FDD tool in controlling
628 the equipment. Any loss of connectivity may cause delays in the control logic with negative
629 consequences on occupant comfort and equipment safety. For example, partner 2 developed two
630 new features "value validation" and "command expiration" to accommodate the asynchronous
631 interaction of the data collection and the auto-correction devices.

632 **6. Conclusion and Future Work**

633 This paper presents the field study of seven fault auto-correction algorithms implemented in
634 commercial FDD platforms. It puts the algorithms in their logical contexts, summarizes their
635 objectives, describes the testing procedure, and shows the successful testing results. The
636 algorithms automatically correct faults and improve the operation of large built-up HVAC systems,
637 focusing on incorrectly programmed schedules, override not released, control hunting, rogue zone,
638 and suboptimal setpoints in HVAC systems. These algorithms were integrated into two
639 commercial FDD platforms and deployed across four buildings and three different building
640 automation systems. The modifications of the FDD tool and the building BAS for auto-correction
641 are summarized in the paper, including FDD-BAS infrastructure update and other software
642 modifications. Each of the seven correction routines was tested in one or two buildings following
643 a rigorous procedure. In general, the enhanced FDD tools were able to correct faults successfully
644 without negatively impacting the system and building occupants. The control hunting correction
645 was tested in a semi-automated way and the schedule correction was successful after some
646 adjustments to the algorithms. Technology benefits, market drivers, and scalability changes are
647 also discussed based on implementation and field testing results, as well as interviews with the
648 FDD providers and facility managers.

649 Future work will focus on more field testing of the auto-correction algorithms with additional FDD
650 platforms in a larger cohort of buildings to prove their robustness. This will include the evaluation
651 of the technical efficacy and the performance of each correction routine, the evaluation of the
652 operations and maintenance benefits for each site in the cohort and the characterization of
653 challenges and best practices. A second area of future work should enhance the auto-correction
654 interface of these FDD tools. This is needed to overcome the natural concerns among end-users
655 about accountability and loss of control, when the FDD routines correct BAS control parameters
656 automatically. At last, the current testing of the auto-correction algorithms was decoupled from
657 the FDD algorithms embedded in the FDD tools. Faults were artificially induced to validate the
658 correction capability. In future, the FDD and auto-correction process needs to be tested together
659 to mitigate the impact of false positives during fault detection.

660 **7. Acknowledgements**

661 This work was supported by the Assistant Secretary for Energy Efficiency and Renewable Energy,
662 Building Technologies Office, of the U.S. Department of Energy under Contract No. DE-AC02-
663 05CH11231. The authors wish to acknowledge Harry Bergmann and Hannah Debelius for their

664 guidance and support of the research. We also thank the fault detection and diagnostics
665 technology and service providers who participated in this study.

666 8. References

- 667 American Society of Heating Refrigerating and Air Conditioning Engineers (ASHRAE). (2018).
668 *Guideline 36–2018. High Performance Sequences of Operation for HVAC Systems.*
669 ASHRAE. Akron, OH, USA, 2018.
- 670 American Society of Heating Refrigerating and Air Conditioning Engineers (ASHRAE). (2020).
671 *Standard 135-2020—BACnet™—A Data Communication Protocol for Building Automation*
672 *and Control Networks.* Available online: [https://www.ashrae.org/technical-](https://www.ashrae.org/technical-resources/bookstore/bacnet)
673 [resources/bookstore/bacnet](https://www.ashrae.org/technical-resources/bookstore/bacnet) (accessed on 10 Aug 2021).
- 674 Benghea, S.C., Li, P., Sarkar, S., Vichik, S., Adetola, V., Kang, K., Lovett, T., Leonardi, F., Kelman,
675 A.D. (2015). Fault-tolerant optimal control of a building HVAC system. *Science and*
676 *Technology for the Built Environment*, 21(6), 734–751.
677 <https://doi.org/10.1080/23744731.2015.1057085>
- 678 Brambley, M., Fernandez, N., Wang, W., Cort, K.A., Cho, H., Ngo, H., Goddard, J.K. (2011). *Final*
679 *project report: Self-correcting controls for VAV system faults filter/fan/coil and VAV box*
680 *sections.* No. PNNL-20452; Pacific Northwest, National Laboratory (PNNL), Richland, WA,
681 USA.
- 682 Dexter A., J. Pakanen (Eds.). (2001) *Demonstrating Automated Fault Detection and Diagnosis*
683 *Methods in Real Buildings*, Technical Research Centre of Finland, Finland.
- 684 Economidou M. (2011). Europe’s buildings under the microscope. A country-by-country review of
685 the energy performance of buildings. Technical Report Buildings Performance Institute
686 Europe.
- 687 Fernandez, N.; Brambley, M.; Katipamula, S. (2009a). *Self-correcting HVAC controls: Algorithms*
688 *for sensors and dampers in air-handling units*, PNNL-19104; Pacific Northwest; National
689 Laboratory: Richland, WA, USA.
- 690 Fernandez, N.; Brambley, M.; Katipamula, S.; Cho, H.; Goddard, J.; Dinh, L. (2009b). *Self-*
691 *correcting HVAC controls project final report*, PNNL-19074; Pacific Northwest; National
692 Laboratory: Richland, WA, USA.
- 693 Fernandez, N.E., Katipamula, S., Wang, W., Xie, Y., Zhao, M. Corbin, C.D. (2017). *Impacts on*
694 *commercial building controls on energy savings and peak load reduction.* Pacific Northwest
695 National Laboratory. PNNL Report Number PNNL-25985, Richland, WA, USA.
- 696 Granderson, J., Singla, R., Mayhorn, E., Ehrlich, P., Vrabie, D., Frank, S. (2017) . *Characterization*
697 *and survey of automated fault detection and diagnostics tools.* Report Number LBNL-
698 2001075.
- 699 Granderson J, Lin G, Singla R, Mayhorn E, Ehrlich P, Vrabie D, Frank S. (2018). Commercial fault
700 detection and diagnostics tools: What they offer, how they differ, and what’s still needed.
701 *2018 ACEEE summer study on energy efficiency in buildings, 12-17 August, 2018, Pacific*
702 *Grove, CA U.S.* <https://escholarship.org/uc/item/4j72k57p>
- 703 Hao, X., Zhang, G., Chen, Y. (2005). Fault-tolerant control and data recovery in HVAC monitoring
704 system. *Energy and Buildings*, 37(2), 175–180.
705 <https://doi.org/10.1016/j.enbuild.2004.06.023>

- 706 Holmberg, D. (2003). *BACnet Wide Area Network Security Threat Assessment*, NIST
707 Interagency/Internal Report (NISTIR), National Institute of Standards and Technology,
708 Gaithersburg, MD, [online], <https://doi.org/10.6028/NIST.IR.7009> (Accessed November 10,
709 2021)
- 710 Johnson Controls Inc, (2020). Help files.
- 711 Kramer, H., Lin, G., Curtin, C., Crowe, E., Granderson, J. (2020). *Proving the business case for*
712 *building analytics*. Lawrence Berkeley National Laboratory.
713 <https://doi.org/10.20357/B7G022>
- 714 Katipamula, S., Brambley, M.R. (2005). Methods for fault detection, diagnostics, and prognostics
715 for building systems—A review, part I. *HVAC and R Research*, 11(1), 3–25.
- 716 Kim, W., Katipamula, S. (2018). A review of fault detection and diagnostics methods for building
717 systems, *Science and Technology for the Built Environment*, 24 (1), 3–21.
718 <https://doi.org/10.1080/23744731.2017.1318008>
- 719 Lin G., Pritoni M., Chen Y., Granderson J. (2020a). Development and implementation of fault-
720 correction algorithms in fault detection and diagnostics tools. *Energies*, 13(10).
721 <https://doi.org/10.3390/en13102598>
- 722 Lin, G, Pritoni M, Chen Y, Moromisato R, Kozlen S, Granderson J. (2020b). Can we fix it
723 automatically? Development of fault auto-correction algorithms for HVAC and lighting
724 systems. *2020 ACEEE Summer Study on Energy Efficiency in Buildings*. 17-21 August,
725 2020, Pacific Grove, CA U.S. <https://doi.org/10.20357/B74C7N>
- 726 Lin, G, Pritoni M, Chen Y, Moromisato R, Kozlen S, Granderson J. (2021). Fault “auto correction”
727 for HVAC systems, a preliminary study. *6th International High Performance Buildings*
728 *Conference*. 24-28 May, 2021, West Lafayette, IN U.S.
729 <https://docs.lib.purdue.edu/ihpbc/369/>
- 730 Peacock M., Johnstone M.N., Valli C. (2018). An exploration of some security issues within the
731 BACnet protocol. In: Mori P., Furnell S., Camp O. (eds) *Information Systems Security and*
732 *Privacy*. ICISSP 2017. *Communications in Computer and Information Science*, 867.
733 Springer, Cham. February 19-21, 2017, Porto, Portugal. [https://doi.org/10.1007/978-3-319-](https://doi.org/10.1007/978-3-319-93354-2_12)
734 [93354-2_12](https://doi.org/10.1007/978-3-319-93354-2_12)
- 735 Pruna E., Sasig E. R., Mullo S., (2017). PI and PID controller tuning tool based on the lambda
736 method, *2017 CHILEAN Conference on Electrical, Electronics Engineering, Information and*
737 *Communication Technologies, CHILECON 2017 - Proceedings, 18-20 Oct. 2017*. Pucon,
738 Chile. <https://doi.org/10.1109/CHILECON.2017.8229616>
- 739 Roth, K.W., Westphalen, D., Feng, M. Y., Llana, P., Quartararo, L. (2005). *Energy impact of*
740 *commercial building controls and performance diagnostics: market characterization, energy*
741 *impact of building faults and energy savings potential*. Report prepared by TIAX LLC for the
742 U.S. Department of Energy.
- 743 Wang, S.; Chen, Y. (2002). Fault-tolerant control for outdoor ventilation air flow rate in buildings
744 based on neural network. *Building and Environment*, 37(7), 691–704.
745 [https://doi.org/10.1016/S0360-1323\(01\)00076-2](https://doi.org/10.1016/S0360-1323(01)00076-2)

- 746 Wang, Z., Wang, Z., He, S., Gu, X., Yan, Z.F. (2017). Fault detection and diagnosis of chillers
747 using Bayesian network merged distance rejection and multi-source non-sensor information.
748 *Applied Energy*, 188, 200-14. <https://doi.org/10.1016/j.apenergy.2016.11.130>
- 749 Wetter, M., Ehrlich, P., Gautier, A., Grahovac, M., Haves, P., Hu, J., Prakash, A., Robin, D. and
750 Zhang, K. (2022). OpenBuildingControl: Digitizing the control delivery from building energy
751 modeling to specification, implementation and formal verification. *Energy*, 238, 121501.
752 <https://doi.org/10.1016/j.energy.2021.121501>
- 753 Wireman, T. (1994). Computerized maintenance management systems. Industrial Press Inc.
- 754 Zhang, Y., Jiang, J. (2008). Bibliographical review on reconfigurable fault-tolerant control systems.
755 *Annual Reviews in Control*, 32(2), 229–252. <https://doi.org/10.1016/j.arcontrol.2008.03.008>
- 756 Zhao Y, Wang S, Xiao F. (2013). Pattern recognition-based chillers fault detection method using
757 support vector data description (SVDD). *Applied Energy*, 112,1041-8.
758 <https://doi.org/10.1016/j.apenergy.2012.12.043>