analytical chemistry

pubs.acs.org/ac Technical Note

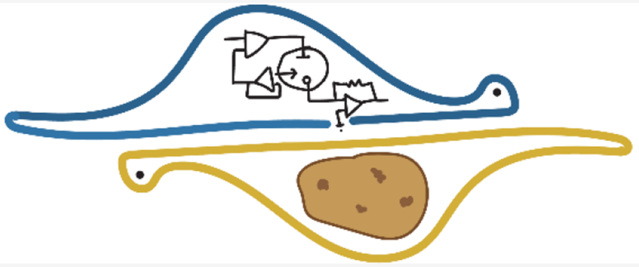
Hard Potato: A Python Library to Control Commercial Potentiostats and to Automate Electrochemical Experiments

Oliver Rodríguez,* Michael A. Pence, and Joaquín Rodríguez-López*

Cite This: https://doi.org/10.1021/acs.analchem.2c04862 Read Online

ACCESS | Metrics & More | Article Recommendations | Supporting Information

ABSTRACT: Here, we develop and show the use of an open-source Python library to control commercial potentiostats. It standardizes the commands for different potentiostat models, opening the possibility to perform automated experiments independently of the instrument used. At the time of this writing, we have included potentiostats from CH Instruments (models 1205B, 1242B, 601E, and 760E) and PalmSens (model Emstat Pico), although the open-source nature of the library allows for more to be included in the future. To showcase the general workflow and implementation of a real experiment, we have automated the Randles−Ševčík methodology to determine the diffusion coefficient of a redox-active species in solution using cyclic voltammetry. This was accomplished by writing a Python script that includes data acquisition, data analysis, and simulation. The total run time was 1 min and 40 s, well below the time it would take even an experienced electrochemist to apply the methodology in a traditional manner. Our library has potential applications that expand beyond the automation of simple repetitive tasks; for example, it can interface with peripheral hardware and well-established third-party Python libraries as part of a more complex and intelligent setup that relies on laboratory automation, advanced optimization, and machine learning.



Electrochemical experimentation is still mostly done in a manual way by introducing parameters one by one into the graphical user interface (GUI) software that comes with the instrument. For general usage, this is arguably enough; however, with recent developments in applications such as redox flow batteries[1] or electrocatalysis,[2] where large amounts of new molecules and materials need to be electrochemically characterized, manual experimentation is cumbersome. Some of the well-known electrochemical instrumentation companies provide some sort of automation capabilities for their potentiostats, either within their own desktop applications via macros and block programming or by an application programming interface (API) written in a general programming language such as C/C++, C#, or Python. APIs can be seen as an intermediary communication step between the user and the instrument, where the user can write simple commands and the API will internally translate them into the set of instructions required by the instrument. This simplifies the syntax on the user side, as they do not need to fully understand the specific details of their potentiostats. APIs are most useful for experienced users, as it allows them to write programs or scripts that include the experiment and to control other instruments and data analysis by means of third-party libraries. To the best of our knowledge at the time of this writing, only PalmSens and Gamry provide frameworks to control their devices via a general-use programming language like Python.[3,4] Other do-it-yourself potentiostats are also reported in the literature, based mostly on open hardware microcontrollers with their source code freely available and can be directly programmed.[5,6] These frameworks, however, lack consistency between each other; this is expected, as each instrument has their own implementations and requirements. Here, we overcome the lack of consistency between different frameworks by introducing Hardware Potentiostato (or Hard Potato for short), a generalized Python API that internally translates high level functions into the specific commands required by each instrument. The selection of Python as the programming language for the library responds to its gentle learning curve, its syntax that is easy to understand, and the sheer number of third-party libraries available to perform data analysis, instrumental control, and interfacing with the operating system. The flexibility given by Python allows users to include electrochemical experimentation onto complex setups, for example, as part of a high throughput system where multiple parameters need to be explored through control of peripheral hardware or when using advanced optimization algorithms and machine learning for autonomous operation. Furthermore, the open-source nature of the library would allow the electrochemistry community to be involved in its develop-

Received: November 2, 2022
Accepted: February 2, 2023

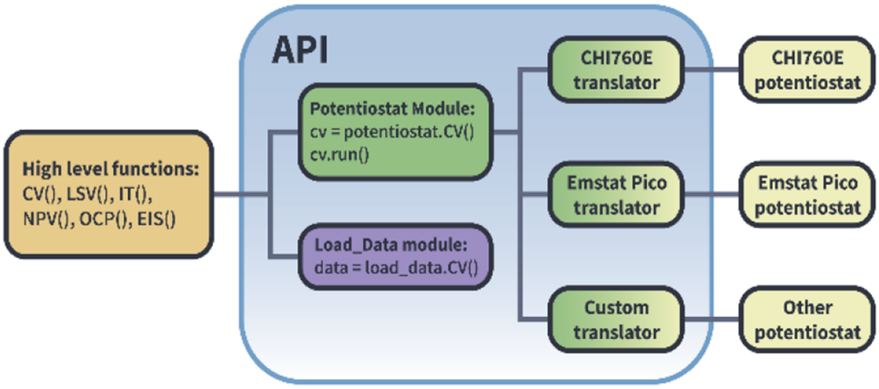


Figure 1. API to control commercial potentiostats. The user writes a script with the high-level functions and the API translates them into the specific commands expected by the instrument.

ment, for example, to include the control of potentiostat models not currently available to the authors.

To test our Python API, we demonstrate its application to the Randles−Ševčík methodology[7] to determine a diffusion coefficient (D) for a soluble redox-active species undergoing electron transfer at an electrode using the cyclic voltammetry (CV) technique. This example is a staple of electrochemical studies, but its implementation promptly requires automation when dealing with large libraries of materials. Furthermore, to showcase the flexibility provided by Python, we make use of Soft Potato,[8,9] an open-source library that contains electrochemical equations and can simulate electrochemical processes assuming semi-infinite planar diffusion and Butler−Volmer kinetics.

EXPERIMENTAL SECTION

API implementation. The API consists of two main components: the potentiostat module and a load_data module. The potentiostat module contains high level functions and calls a "translator" for each commercial potentiostat available in the library, while the load_data module provides easy data handling (see Figure 1). The potentiostat module provides generalized functions for typical electrochemical techniques; for example, some of the arguments for the potentiostat.CV() function are the scan rate (sr [V/s]), the first vertex potential (Ev1 [V]), and the second vertex potential (Ev2 [V]). The Supporting Information includes instructions to install the library (Section S1) and script examples for typical experiments as well as the Randles−Ševčík methodology implemented here (Sections S3−S8). (See refs 10 and 11 for the latest version of the library and full documentation.)

Figure 1 shows how a common API facilitates communication between the user and each instrument. The generalized functions act as intermediaries between the user and the instrument, where the user does not have to worry about the specific ways on which their commercial software defines the parameters. Internally, these functions are "translated" into specific commands that each instrument understands. In the current version of the library (v1.3.11),[12] we have included several CH Instruments potentiostats. For the following, we are using the CHI760E, a highly popular and versatile research grade potentiostat from CH Instruments with the capability to be controlled via internal macro commands, but not directly with Python. The CHI software allows one to save the instructions as a .mcr text file and to use it as a parameter when calling the software from the Windows command line. The software then starts and runs the instructions written on the macro file. Our library uses this to control the potentiostat, where the user only needs to write the high-level functions; then, internally, the API converts these instructions into macro commands that the CHI760E understands. The resulting macro file also contains instructions to save the data into a .txt and .bin file, the latter being the format expected by the CHI software. It also contains a termination line that closes the software and returns control to Python to continue with the next instruction line.[13] While we have focused on the CHI760E, it is possible to include more potentiostats into the library as depicted in Figure 1, where a developer would only need to write its "translator" to match the API's functions to the instrument of interest. Until January 2023, we have written the translators for the CH Instrument potentiostats models 1205B, 1242B, 601E, and 760E and the PalmSens Emstat Pico potentiostat, the latter enabling the use of a low-cost potentiostat for application in an automated electrochemical platform.

Controlling instruments from different vendors comes with certain challenges. For example, the CHI760E can perform automated iR compensation if a resistance value is provided, while this is not possible with the Emstat Pico. Meanwhile, the Emstat Pico has different operation modes (low speed, high speed, and maximum potential range) that can be selected. To consider the variability in options for each model, we have included optional parameters specific to each instrument. For example, in cyclic voltammetry, a resistance value can be included if using the CHI760E; this value will be ignored if using the Emstat Pico. This also allowed us to future-proof the code, as developers can write the translators for their own instruments including the core parameters required by the general library and optional parameters only used by their specific instrument. We have also included a validation step to ensure that potential parameters for each technique and potentiostat model fall within their corresponding range. For more details on the inner workings of the library, see refs 10−12.

In the following sections, we use the CH Instruments 760E to demonstrate how the library can be used to automate data acquisition and include data analysis and simulation on the same script. For this, we have selected the Randles−Ševčík routine, a highly used methodology and part of the general electrochemistry toolkit that consists of multiple repetitions and data analysis that may become tedious. To compare between different potentiostat models and to showcase the flexibility of Hard Potato, we have included the results using the CHI760E, CHI1205B, and Emstat Pico and their respective Python scripts

https://doi.org/10.1021/acs.analchem.2c04862

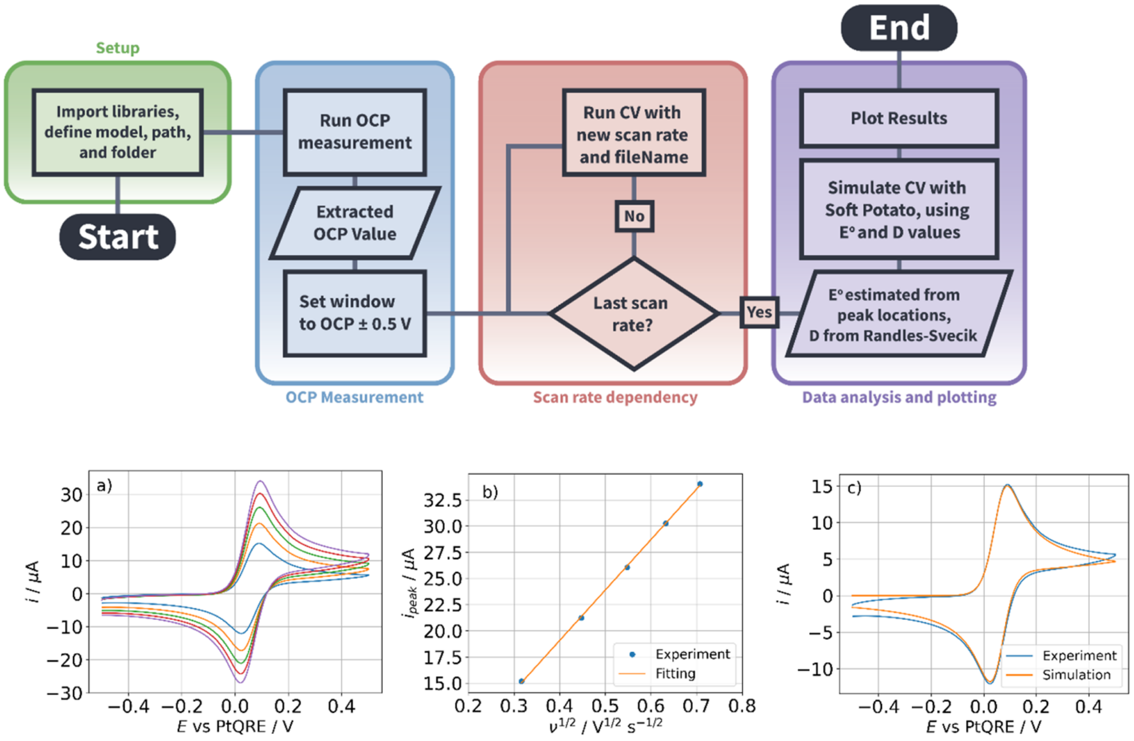


Figure 2. Results of the automated Randles−Ševčík routine execution. Top: Block diagram of the routine. Bottom: (a) cyclic voltammograms performed at 0.1, 0.2, 0.3, 0.4, and 0.5 V/s from bottom to top, (b) linear regression following eq 1 to estimate D, and (c) comparison between a cyclic voltammogram recorded at 0.1 V/s and its simulated response. The solution used consisted of a mixture of 1 mM FcMeOH and 100 mM KNO_3. The simulation was performed assuming planar diffusion and the Butler−Volmer equation with fast electron transfer kinetics; the parameters used were n = 1, $k^0 = 1 \times 10^8$ cm/s, and $\alpha = 0.5$.

in the Supporting Information, Section S8, and as a Supporting Information zip file.

Chemicals and methods. Hydroxymethylferrocene (FcMeOH) (Sigma-Aldrich, 97%) and potassium nitrate (Sigma-Aldrich reagent grade) were used as purchased. A three-electrode electrochemical cell was used with a Pt disc (a = 1 mm, CH Instruments) as a working electrode and two Pt wires used as counter and quasi-reference electrodes. A CH Instruments potentiostat model 760E was used to perform the electrochemical experiments. The Python script was run on a MS Windows 8.1 computer with Python version 3.10.6 installed. The libraries used were hard potato1.3.11,[10] numpy 1.23.1 for data analysis,[14] scipy 1.9.0 for linear regression,[15] matplotlib 3.5.2 for plotting the data,[16] and softpotato 1.0.18 for the electrochemical simulations.[8,9]

RESULTS

Randles−Ševčík is one of the most used electroanalytical methodologies. It allows one to measure the number of electrons, the geometrical area of the electrode, or the diffusion coefficient of a redox species, provided the rest of the electrochemical parameters are known. It is also one of the first methodologies taught in introductory electrochemistry courses.[17] It consists of performing CV using a macroelectrode with a redox active species and analyzing the peak currents as a function of the square root of the scan rate. If the process is diffusion controlled and displays fast electron transfer kinetics, a linear relationship is observed that is described by the Randles−Ševčík equation:

$$i_p = 0.4463nFAC\left(\frac{nF\nu D}{RT}\right)^{1/2} \quad (1)$$

where i_p is the peak current, n is the number of electrons, F is the Faraday constant (96,485 C/mol), C is the bulk concentration of the species, A is the geometrical area of the electrode, D is the diffusion coefficient, ν is the scan rate, R is the gas constant, and T is the temperature. The repetitive nature of the Randles−Ševčík methodology, with the variation of one parameter (ν) and the estimation of another (D), makes it a perfect candidate for automation. Figure 2 shows a block diagram of the implementation. Briefly, the routine consists of four main parts. In the first part, we declare the variables required by the setup, which are the potentiostat model, path to the .exe program in the case of the CHI760E, and the folder where to save the data. Then, in the second part, we perform an open circuit potential (OCP) measurement and use it to estimate the potential window of the oxidation of FcMeOH; this could be useful when standard reduction potential of the species (E^0) is unknown, for example, when characterizing new nonaqueous redox molecules with unknown redox potentials. The third part consists of a loop that describes the specific steps during measurement, one for each scan rate. Here, we also import the recently acquired data to be used for the analysis later. The fourth part consists of the data analysis and plotting, that is, the estimation of E^0 from the average of the forward and reverse peak potentials and D from eq 1. Importantly, this step allows us to interface with other Python libraries that are useful for electrochemistry. For instance, in our analysis, we used the Randles−Ševčík equation included in the Soft Potato library to fit the experimental peak currents and obtain D. We performed

https://doi.org/10.1021/acs.analchem.2c04862

linear regression to obtain the slope of the i_{peak} vs $\nu^{1/2}$ curve and used the R^2 value to ensure our experiments were above a certain threshold, otherwise meaning that the methodology would be inappropriate to use due to either experimental artifacts (resistance, capacitance, noise), the effect of slow kinetics, or the presence of multiple electron transfer steps. Adding controls such as these prevents the need for manual inspection of the data, which would defeat the purpose of having a fully automated system, or when protocols are used by nonexpert electrochemists. We have also used Soft Potato to simulate the cyclic voltammograms once E^0 and D were estimated. Using Soft Potato in this case helps to confirm the accuracy of the entire approach by plotting the experimental results alongside their simulations. Figure 2a −c shows the results of the automated Randles−Ševčík experimentation, data analysis, and comparison to simulation, promptly displaying the obtained CVs, the linear relationship predicted by eq 1, and the comparison between an experimental and simulated voltammogram. The total time of execution of this script was 1 min and 40 s, well below the total time it would take even an experienced electrochemist to apply the full methodology manually. To this point, the first author of this work tested it themselves taking ∼30 min.

■ DISCUSSION

The use of our API simplifies the execution of straightforward methods such as CV and enables the automation of electrochemical experiments. While the execution time in the case shown here depends on the scan rates used, performing the full methodology in a traditional way is still time consuming as it requires changing the scan rate manually for each CV, saving data in an appropriate format to be used in commercial data analysis software, extracting E^0 and peak currents (or other suitable parameters), and performing linear regression against the square root of the scan rate to obtain D. Additionally, the user could still invest time in manually inputting parameters to perform electrochemical simulations, export the simulated data, and put it back into the data analysis software to compare it with the experimental data. This methodology is straightforward and can be carried out manually; however, its full implementation from experiment to data analysis and simulation is tedious and time consuming. Our API overcomes practical obstacles not only by offering automation of all these functions, but also by introducing a consistent language that defines commonly used parameters that are independent of the instrument used.

The API approach shown here also raises new prospects for combining electrochemical methods with a wide variety of Python libraries. While we focused on the Randles−Ševčík methodology, basically, any other electrochemical routine can be easily automated. For example, immediate systematic experiments similar to the one here tested are found in electrochemical applications such as potential window opening studies,[18] the Koutecky−Levich analysis,[19] the implementation of sampled current voltammetry,[20−22] the measurement of electrochemical kinetics with macro[23] and microelectrodes,[24] capacitance and solution resistance measurements with voltammetry and electrochemical impedance spectroscopy,[25,26] and the electrochemistry part of a homemade scanning electrochemical microscope.[27] For example, in our group, we have recently reported the use of automated tools to estimate chemical decomposition rates using a set of interdigitated electrode arrays and an early version of Hard Potato.[28] The Hard Potato library enables researchers to automate these tasks, especially when dealing with repetition and/or characterization of large libraries of redox species.[29] The ability to interface with third-party libraries for data analysis, plotting, simulations, machine learning, and even other instruments such as spectrophotometers and (micro)fluidics makes this a very powerful tool. For example, here, we used Soft Potato, an open-source Python library that includes typical electrochemical equations and can perform simulations assuming semi-infinite planar diffusion and Butler−Volmer kinetics. Soft Potato makes the calculation of D from the Randles−Ševčík equation straightforward, as the user does not need to manually code it. We also used it to simulate the electrochemical response and compare it to the corresponding experiment, everything on the same script and immediately after the experiments were performed. Fully integrated, the Potato ecosystem can become a powerful tool to automate electrochemical experimentation and data analysis. Furthermore, a script can be shared between research groups "as is" to corroborate results or to make changes to the methodology, supporting the philosophy of more open and transparent analytical science. The potential for this approach can be significantly enhanced if companies offer compatible Python frameworks and the research community is involved with the development and improvement of the library. Specific challenges that can be addressed are the ability to live stream data to perform operations while the measurement is being taken, for example, to change current ranges, detect oscillations due to *iR* compensation, and even emergency stops. We believe that this will be essential in the coming years, since we see automation making its way into the chemistry realm, specifically, synthetic chemistry, where groups around the world are building fully automated synthesis laboratories.[30,31] It is only a matter of time until digital electrochemistry becomes the norm.

■ CONCLUSIONS

Here, we have demonstrated the flexibility of automated electrochemical experimentation enabled by our open-source Hard Potato Python library. As a use case, we have implemented a workflow to estimate E^0 from voltammograms and D from the Randles−Ševčík methodology, using these parameters to simulate and corroborate the electrochemical response, everything on a single script. The total run time was 1 min and 40 s, limited only by the scan rates used and the number of experiments performed, as the data analysis and simulation are virtually instantaneous. The ability to centralize data acquisition, analysis, and simulation within a single Python script showcases how powerful and flexible this tool is. Our primary goal is to automate processes and facilitate the sharing of electrochemical protocols, as groups can run the same experiments independently of the potentiostat that they possess, provided they have access to the Python script and the potentiostat "translator" is included in the library. Overall, we believe that our generalized Hard Potato library can be incorporated into any electrochemistry workflow and opens the possibility to perform more interesting problems involving automation, high throughput experimentation, and even artificial intelligence for closed-loop experimental design.

■ ASSOCIATED CONTENT

Supporting Information

The Supporting Information is available free of charge at https://pubs.acs.org/doi/10.1021/acs.analchem.2c04862.

Instructions on how to install the Hard Potato library, Python script examples to perform cyclic voltammetry, chronoamperometry, open circuit potential measurement, linear sweep voltammetry, and the Randles−Ševčík methodology performed with a CHI760E, CHI1205B, and Emstat Pico (PDF)

Screen recording of the automation of the Randles−Ševčík methodology (MP4)

Data (ZIP)

■ AUTHOR INFORMATION

Corresponding Authors

Oliver Rodríguez − *Department of Chemistry, University of Illinois at Urbana−Champaign, Urbana, Illinois 61801, United States; Beckman Institute for Advanced Science and Technology, University of Illinois at Urbana−Champaign, Urbana, Illinois 61801, United States; Joint Center for Energy Storage Research (JCESR), Argonne National Laboratory, Lemont, Illinois 60439, United States;* orcid.org/0000-0003-2350-1214; Email: oliver.rdz@softpotato.xyz

Joaquín Rodríguez-López − *Department of Chemistry, University of Illinois at Urbana−Champaign, Urbana, Illinois 61801, United States; Beckman Institute for Advanced Science and Technology, University of Illinois at Urbana−Champaign, Urbana, Illinois 61801, United States; Joint Center for Energy Storage Research (JCESR), Argonne National Laboratory, Lemont, Illinois 60439, United States;* orcid.org/0000-0003-4346-4668; Email: joaquinr@illinois.edu

Author

Michael A. Pence − *Department of Chemistry, University of Illinois at Urbana−Champaign, Urbana, Illinois 61801, United States; Beckman Institute for Advanced Science and Technology, University of Illinois at Urbana−Champaign, Urbana, Illinois 61801, United States; Joint Center for Energy Storage Research (JCESR), Argonne National Laboratory, Lemont, Illinois 60439, United States;* orcid.org/0000-0001-5880-9812

Complete contact information is available at:
https://pubs.acs.org/10.1021/acs.analchem.2c04862

Notes

The authors declare no competing financial interest.

■ ACKNOWLEDGMENTS

The research was financially supported by the Joint Center for Energy Storage Research (JCESR), an Energy Innovation Hub funded by the U.S. Department of Energy, Office of Science, Basic Energy Sciences. The authors would like to acknowledge the reviewers for their useful suggestions that made this work stronger, specifically, the ideas to future-proof the code to allow for the variability in potentiostat models and the highlighting of Soft Potato as part of the Potato ecosystem.

■ REFERENCES

(1) Sevov, C. S.; Brooner, R. E. M.; Chénard, E.; Assary, R. S.; Moore, J. S.; Rodríguez-López, J.; Sanford, M. S. *J. Am. Chem. Soc.* **2015**, *137* (45), 14465−14472.

(2) Wang, G.; Li, W.; Huang, B.; Xiao, L.; Lu, J.; Zhuang, L. *ACS Appl. Energy Mater.* **2019**, *2* (5), 3160−3165.

(3) *PalmSens MethodSCRIPT*. https://www.palmsens.com/methodscript/ (accessed 2022−09−28).

(4) *Gamry Instruments Software*. https://www.gamry.com/support-2/software/ (accessed 2022−09−28).

(5) Meloni, G. N. *J. Chem. Educ.* **2016**, *93* (7), 1320−1322.

(6) Brown, E. W.; Glasscott, M. W.; Conley, K.; Barr, J.; Ray, J. D.; Moores, L. C.; Netchaev, A. *Anal. Chem.* **2022**, *94* (12), 4906−4912.

(7) Bard, A. J.; Faulkner, L. R. *Electrochemical Methods. Fundamentals and Applications*, Second ed.; John Wiley & Sons, Inc.: New York, 2001.

(8) Rodríguez, O. Soft Potato: An Open-Source Electrochemical Simulator and Toolkit, *ChemRxiv Preprint*, 2022. DOI: 10.26434/chemrxiv-2022-4bs3w.

(9) Rodríguez, O. *Soft Potato: an open-source electrochemistry toolkit*. https://github.com/oliverrdz/SoftPotato.

(10) *Hard Potato GitHub*. https://github.com/jrlLAB/hardpotato (accessed 2022−09−28).

(11) *Hard Potato Wiki Page*. https://github.com/jrlLAB/hardpotato/wiki.

(12) *Hard Potato source code v1.3.11*. https://github.com/jrlLAB/hardpotato/releases/tag/v1.3.11.

(13) The implementation of the termination command in the CHI760E requires having the latest firmware and software installed. Contact the CH Instruments Support to update them.

(14) Harris, C. R.; Millman, K. J.; van der Walt, S. J.; Gommers, R.; Virtanen, P.; Cournapeau, D.; Wieser, E.; Taylor, J.; Berg, S.; Smith, N. J.; Kern, R.; Picus, M.; Hoyer, S.; van Kerkwijk, M. H.; Brett, M.; Haldane, A.; del Río, J. F.; Wiebe, M.; Peterson, P.; Gérard-Marchant, P.; Sheppard, K.; Reddy, T.; Weckesser, W.; Abbasi, H.; Gohlke, C.; Oliphant, T. E. *Nature* **2020**, *585*, 357−362.

(15) Virtanen, P.; Gommers, R.; Oliphant, T. E.; Haberland, M.; Reddy, T.; Cournapeau, D.; Burovski, E.; Peterson, P.; Weckesser, W.; Bright, J.; van der Walt, S. J.; Brett, M.; Wilson, J.; Millman, K. J.; Mayorov, N.; Nelson, A. R. J.; Jones, E.; Kern, R.; Larson, E.; Carey, C. J.; Polat, İ.; Feng, Y.; Moore, E. W.; VanderPlas, J.; Laxalde, D.; Perktold, J.; Cimrman, R.; Henriksen, I.; Quintero, E. A.; Harris, C. R.; Archibald, A. M.; Ribeiro, A. H.; Pedregosa, F.; van Mulbregt, P.; Vijaykumar, A.; Bardelli, A. P.; Rothberg, A.; Hilboll, A.; Kloeckner, A.; Scopatz, A.; Lee, A.; Rokem, A.; Woods, C. N.; Fulton, C.; Masson, C.; Häggström, C.; Fitzgerald, C.; Nicholson, D. A.; Hagen, D. R.; Pasechnik, D. v.; Olivetti, E.; Martin, E.; Wieser, E.; Silva, F.; Lenders, F.; Wilhelm, F.; Young, G.; Price, G. A.; Ingold, G. L.; Allen, G. E.; Lee, G. R.; Audren, H.; Probst, I.; Dietrich, J. P.; Silterra, J.; Webber, J. T.; Slavič, J.; Nothman, J.; Buchner, J.; Kulick, J.; Schönberger, J. L.; de Miranda Cardoso, J. V.; Reimer, J.; Harrington, J.; Rodríguez, J. L. C.; Nunez-Iglesias, J.; Kuczynski, J.; Tritz, K.; Thoma, M.; Newville, M.; Kümmerer, M.; Bolingbroke, M.; Tartre, M.; Pak, M.; Smith, N. J.; Nowaczyk, N.; Shebanov, N.; Pavlyk, O.; Brodtkorb, P. A.; Lee, P.; McGibbon, R. T.; Feldbauer, R.; Lewis, S.; Tygier, S.; Sievert, S.; Vigna, S.; Peterson, S.; More, S.; Pudlik, T.; Oshima, T.; Pingel, T. J.; Robitaille, T. P.; Spura, T.; Jones, T. R.; Cera, T.; Leslie, T.; Zito, T.; Krauss, T.; Upadhyay, U.; Halchenko, Y. O.; Vázquez-Baeza, Y. *Nat. Methods* **2020**, *17* (3), 261−272.

(16) Hunter, J. D. *Comput. Sci. Eng.* **2007**, *9* (3), 90−95.

(17) Elgrishi, N.; Rountree, K. J.; McCarthy, B. D.; Rountree, E. S.; Eisenhart, T. T.; Dempsey, J. L. *J. Chem. Educ.* **2018**, *95* (2), 197−206.

(18) Diaz-Morales, O.; Hersbach, T. J. P.; Badan, C.; Garcia, A. C.; Koper, M. T. M. *Faraday Discuss.* **2018**, *210*, 301−315.

(19) Masa, J.; Batchelor-McAuley, C.; Schuhmann, W.; Compton, R. G. *Nano Res.* **2014**, *7* (1), 71−78.

(20) Soares, C. O.; Rodríguez, O.; Buvat, G.; Duca, M.; Garbarino, S.; Guay, D.; Denuault, G.; Tavares, A. C. *Electrochim. Acta* **2020**, *362*, 136946.

(21) Perry, S. C.; al Shandoudi, L. M.; Denuault, G. *Anal. Chem.* **2014**, *86* (19), 9917−9923.

(22) Rodríguez, O.; Denuault, G. *J. Electroanal. Chem.* **2021**, *882*, 115021.

(23) Nicholson, R. S. *Anal. Chem.* **1965**, *37* (11), 1351−1355.

(24) Kim, J.; Bard, A. J. *Anal. Chem.* **2016**, *88* (3), 1742−1747.

(25) Morales, D. M.; Risch, M. *JPhys. Energy* **2021**, *3* (3), 034013.

(26) Ciucci, F. *Current Opinion in Electrochemistry* **2019**, *13*, 132−139.

(27) Meloni, G. N. *Anal. Chem.* **2017**, *89* (17), 8643−8649.

(28) Pence, M. A.; Rodríguez, O.; Lukhanin, N. G.; Schroeder, C. M.; Rodríguez-López, J. *ACS Measurement Science Au* **2022**, DOI: 10.1021/acsmeasuresciau.2c00054.

(29) Hoar, B. B.; Zhang, W.; Xu, S.; Deeba, R.; Costentin, C.; Gu, Q.; Liu, C. *ACS Measurement Science Au* **2022**, *2*, 595.

(30) Langner, S.; Häse, F.; Perea, J. D.; Stubhan, T.; Hauch, J.; Roch, L. M.; Heumueller, T.; Aspuru-Guzik, A.; Brabec, C. J. *Adv. Mater.* **2020**, *32* (14), 1907801.

(31) Li, S.; Jira, E. R.; Angello, N. H.; Li, J.; Yu, H.; Moore, J. S.; Diao, Y.; Burke, M. D.; Schroeder, C. M. *Nat. Commun.* **2022**, *13* (1), 2102.