FYI  
  
----- Forwarded by Anton Dekusar/Ireland/IBM on 23/03/2020 10:57 -----  
  
From: Julien Gacon <notifications@github.com>  
To: stefan-woerner/qiskit-aqua <qiskit-aqua@noreply.github.com>  
Cc: adekusar-drl <adekusar@ie.ibm.com>, Mention <mention@noreply.github.com>  
Date: 23/03/2020 10:20  
Subject: [EXTERNAL] Re: [stefan-woerner/qiskit-aqua] Latest version of the ADMMOptimizer Implementation (#50)

**@Cryoris** requested changes on this pull request.

Hi [@adekusar-drl](https://urldefense.proofpoint.com/v2/url?u=https-3A__github.com_adekusar-2Ddrl&d=DwMFaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=sqbsN76ZNhxb9WaLQsmrvHpMr-81A1A_aEDd-6W-z4w&m=XSEp6YB7H5KwkL344WgK8egaonOg7dRhK9gtVTWU4pY&s=MD9ZHcVHZR8VgUYMnXan-ZoQeSQpVwLlrZNpaCdoZXA&e=), thanks for the updates on the ADMM code.

I've got a few comments before I think this is ready to merge. Most of them are concerning typing, docstrings and variable names, nothing big. 🙂

1. Aqua uses pylint and pycodestyle to ensure that the code fits the general style guidelines. You can check this by running

make lint spell style

Can you make sure that there are no errors or warnings when you run this command?  
Otherwise the files cannot be merged to Aqua.

[Anton]

1. In the code you choose a few variable names that are not very descriptive, e.g. tau and mu or all the matrices (q0,c0,a2,....). If somebody wants to really comprehend the code this will be difficult since there are no comments describing what you actually mean by these variables.  
   This can be resolved by either renaming the variables (which would probably be a lot of work), or adding comments. Even better would be a reference paper or work that introduces all these variable names and where one could look up how the variable is defined.  
   Is there such a paper you could add as reference?

[Claudio: to check whether a report can be published]

1. In general the all non-private methods (i.e. no leading underscore) should have type hints for the arguments and the return type.  
   [Anton]
2. The docstring style used in Qiskit is

"""One-line summary.  
<linebreak>  
More detail.  
<linebreak>  
Args:  
  ... from here on it was always correct in the code.

or, if you only have a single-line summary

"""One-line summary."""

If you have any concerns or remarks I'm happy to discuss them!

[Anton]

In [qiskit/optimization/algorithms/admm\_optimizer.py](https://urldefense.proofpoint.com/v2/url?u=https-3A__github.com_stefan-2Dwoerner_qiskit-2Daqua_pull_50-23discussion-5Fr396291364&d=DwMFaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=sqbsN76ZNhxb9WaLQsmrvHpMr-81A1A_aEDd-6W-z4w&m=XSEp6YB7H5KwkL344WgK8egaonOg7dRhK9gtVTWU4pY&s=8MGz03WJvxmPACF3ny3GErCCFoX9GXJIbQx4nUxFxF8&e=" \t "_blank):

> +                 mu=1000, qubo\_optimizer: OptimizationAlgorithm = None,  
+                 continuous\_optimizer: OptimizationAlgorithm = None) -> None:

If None is supported as a value, this should be annotated in the type hints using the Optional descriptor.  
(This is a shorthand notation for Union[OptimizationAlgorithm, None].)

⬇️ Suggested change  
-                 mu=1000, qubo\_optimizer: OptimizationAlgorithm = None,  
-                 continuous\_optimizer: OptimizationAlgorithm = None) -> None:  
+                 mu=1000, qubo\_optimizer: Optional[OptimizationAlgorithm] = None,  
+                 continuous\_optimizer: Optional[OptimizationAlgorithm] = None) -> None:

[Anton]

In [qiskit/optimization/algorithms/admm\_optimizer.py](https://urldefense.proofpoint.com/v2/url?u=https-3A__github.com_stefan-2Dwoerner_qiskit-2Daqua_pull_50-23discussion-5Fr396291805&d=DwMFaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=sqbsN76ZNhxb9WaLQsmrvHpMr-81A1A_aEDd-6W-z4w&m=XSEp6YB7H5KwkL344WgK8egaonOg7dRhK9gtVTWU4pY&s=oNkJcWo-zgftG1wnetlbB6V19xeGHM4R3QEErosFxeM&e=" \t "_blank):

>   
class ADMMParameters:  
+    """Defines a set of parameters for ADMM optimizer.  
+    """  
+  
    def \_\_init\_\_(self, rho\_initial=10000, factor\_c=100000, beta=1000, max\_iter=10, tol=1.e-4, max\_time=1800,

Type hints are missing here (all arguments should have them, not only the optimizers).

[Anton]

In [qiskit/optimization/algorithms/admm\_optimizer.py](https://urldefense.proofpoint.com/v2/url?u=https-3A__github.com_stefan-2Dwoerner_qiskit-2Daqua_pull_50-23discussion-5Fr396303374&d=DwMFaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=sqbsN76ZNhxb9WaLQsmrvHpMr-81A1A_aEDd-6W-z4w&m=XSEp6YB7H5KwkL344WgK8egaonOg7dRhK9gtVTWU4pY&s=DUS0Ex6nbPa7sPw0Y1o-pk0l1EpDqVPcL6K_6peuO-M&e=" \t "_blank):

> +    """Defines a set of parameters for ADMM optimizer.  
+    """

If the docstring consists only of the single line summary we usually put everything in one line 🙂

⬇️ Suggested change  
-    """Defines a set of parameters for ADMM optimizer.  
-    """  
+    """Defines a set of parameters for ADMM optimizer."""

[Anton]

In [qiskit/optimization/algorithms/admm\_optimizer.py](https://urldefense.proofpoint.com/v2/url?u=https-3A__github.com_stefan-2Dwoerner_qiskit-2Daqua_pull_50-23discussion-5Fr396304878&d=DwMFaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=sqbsN76ZNhxb9WaLQsmrvHpMr-81A1A_aEDd-6W-z4w&m=XSEp6YB7H5KwkL344WgK8egaonOg7dRhK9gtVTWU4pY&s=hWMoo1uLZSB5qq9H8OXicJvNc9EgEGo9ypgrldsvXVg&e=" \t "_blank):

>   
class ADMMParameters:  
+    """Defines a set of parameters for ADMM optimizer.  
+    """  
+  
    def \_\_init\_\_(self, rho\_initial=10000, factor\_c=100000, beta=1000, max\_iter=10, tol=1.e-4, max\_time=1800,  
                 three\_block=True, vary\_rho=0, tau\_incr=2, tau\_decr=2, mu\_res=10,

Should vary\_rho not rather be a bool? Below you're saying If 0, then... and if 1 then ..., in which case using True/False would be clearer

[Anton: vary\_rho may take only two string values, UPDATE\_10\_PERCENT = 0 and UPDATE\_BY\_RESIDUALS=1. Discussed with Claudio on slack]

In [qiskit/optimization/algorithms/admm\_optimizer.py](https://urldefense.proofpoint.com/v2/url?u=https-3A__github.com_stefan-2Dwoerner_qiskit-2Daqua_pull_50-23discussion-5Fr396306026&d=DwMFaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=sqbsN76ZNhxb9WaLQsmrvHpMr-81A1A_aEDd-6W-z4w&m=XSEp6YB7H5KwkL344WgK8egaonOg7dRhK9gtVTWU4pY&s=62VDNcaGbwULmGH7_9Mo2qxuj4I71E4a_DwodOzEz8Q&e=" \t "_blank):

>              tau\_incr: Parameter used in the rho update.  
            tau\_decr: Parameter used in the rho update.  
            mu\_res: Parameter used in the rho update.

These are not very descriptive, could you either describe in slightly more detail what parameter this is (like you did for the parameter mu) or alternatively point to a paper using the names tau and mu so people can retrace where they are used?

[Claudio]

In [qiskit/optimization/algorithms/admm\_optimizer.py](https://urldefense.proofpoint.com/v2/url?u=https-3A__github.com_stefan-2Dwoerner_qiskit-2Daqua_pull_50-23discussion-5Fr396307002&d=DwMFaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=sqbsN76ZNhxb9WaLQsmrvHpMr-81A1A_aEDd-6W-z4w&m=XSEp6YB7H5KwkL344WgK8egaonOg7dRhK9gtVTWU4pY&s=NazXH568MndkhAURUqhDFZfc2nJxvJ3gST3NAOn3zU8&e=" \t "_blank):

> +    """Internal computation state of the ADMM implementation. Here, various variables are stored that are  
+    being updated during problem solving. The values are relevant to the problem being solved.  
+    The state is recreated for each optimization problem. State is returned as the third value  
+    """

The docstyle we try to use throughout Qiskit is

"""One-line summary.  
  
More detail.  
"""  
⬇️ Suggested change  
-    """Internal computation state of the ADMM implementation. Here, various variables are stored that are  
-    being updated during problem solving. The values are relevant to the problem being solved.  
-    The state is recreated for each optimization problem. State is returned as the third value  
-    """  
+    """Internal computation state of the ADMM implementation.  
+      
+    Here, various variables are stored that are  
+    being updated during problem solving. The values are relevant to the problem being solved.  
+    The state is recreated for each optimization problem. State is returned as the third value  
+    """

[Anton]

In [qiskit/optimization/algorithms/admm\_optimizer.py](https://urldefense.proofpoint.com/v2/url?u=https-3A__github.com_stefan-2Dwoerner_qiskit-2Daqua_pull_50-23discussion-5Fr396310363&d=DwMFaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=sqbsN76ZNhxb9WaLQsmrvHpMr-81A1A_aEDd-6W-z4w&m=XSEp6YB7H5KwkL344WgK8egaonOg7dRhK9gtVTWU4pY&s=2ZFA0OlGfYkjWipr4NPAOZCaIyntVEj3snteskd1JZE&e=" \t "_blank):

> @@ -98,6 +100,23 @@ def \_\_init\_\_(self,  
        self.continuous\_indices = continuous\_indices  
        self.sense = op.objective.get\_sense()

What would you think about using a matrix\_cache dictionary for these variables, as:

self.matrix\_cache = {'q0': None, 'c0': None, ...}   # or cache, or matrices, ...

It would be self-explanatory that this is used as cache (good for code readability) and also all the names you chose (q0,c0,...) are invalid names for pylint. Using a dict would resolve that issue too.

[Anton]

In [qiskit/optimization/algorithms/admm\_optimizer.py](https://urldefense.proofpoint.com/v2/url?u=https-3A__github.com_stefan-2Dwoerner_qiskit-2Daqua_pull_50-23discussion-5Fr396310523&d=DwMFaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=sqbsN76ZNhxb9WaLQsmrvHpMr-81A1A_aEDd-6W-z4w&m=XSEp6YB7H5KwkL344WgK8egaonOg7dRhK9gtVTWU4pY&s=1w59lEy4qWLJRhccNvfzIuFQHGzYOlII-qYCZBruiYM&e=" \t "_blank):

> +    """An implementation of the ADMM algorithm.  
+    """  
  
⬇️ Suggested change  
-    """An implementation of the ADMM algorithm.  
-    """  
+    """An implementation of the ADMM algorithm."""

[Anton]

In [qiskit/optimization/algorithms/admm\_optimizer.py](https://urldefense.proofpoint.com/v2/url?u=https-3A__github.com_stefan-2Dwoerner_qiskit-2Daqua_pull_50-23discussion-5Fr396310736&d=DwMFaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=sqbsN76ZNhxb9WaLQsmrvHpMr-81A1A_aEDd-6W-z4w&m=XSEp6YB7H5KwkL344WgK8egaonOg7dRhK9gtVTWU4pY&s=bSO8zj6VwGCySnt7dGIg7BlzfJr8o6lbC2vG-bH-jgk&e=" \t "_blank):

> @@ -121,12 +140,20 @@ def \_\_init\_\_(self,  
  
  
class ADMMOptimizer(OptimizationAlgorithm):  
+    """An implementation of the ADMM algorithm.  
+    """  
+  
    def \_\_init\_\_(self, params: ADMMParameters = None) -> None:

Add Optional type hint

[Anton]

In [qiskit/optimization/algorithms/admm\_optimizer.py](https://urldefense.proofpoint.com/v2/url?u=https-3A__github.com_stefan-2Dwoerner_qiskit-2Daqua_pull_50-23discussion-5Fr396311357&d=DwMFaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=sqbsN76ZNhxb9WaLQsmrvHpMr-81A1A_aEDd-6W-z4w&m=XSEp6YB7H5KwkL344WgK8egaonOg7dRhK9gtVTWU4pY&s=ogcmyW8aAastg0o7_Y2gwGwW4ry9PL8Cvj2JTxWLmTU&e=" \t "_blank):

>          if params is None:  
            # create default params  
            params = ADMMParameters()

You could use the single line version here

⬇️ Suggested change  
-        if params is None:  
-            # create default params  
-            params = ADMMParameters()  
+        params = params or ADMMParameters()  # create default params if needed

[Anton]

In [qiskit/optimization/algorithms/admm\_optimizer.py](https://urldefense.proofpoint.com/v2/url?u=https-3A__github.com_stefan-2Dwoerner_qiskit-2Daqua_pull_50-23discussion-5Fr396311915&d=DwMFaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=sqbsN76ZNhxb9WaLQsmrvHpMr-81A1A_aEDd-6W-z4w&m=XSEp6YB7H5KwkL344WgK8egaonOg7dRhK9gtVTWU4pY&s=75MFr3XpJuDL3LxoYmu5qUqf-t_tB7W_YU8VzD2jtUk&e=" \t "_blank):

>          super().\_\_init\_\_()  
        if params is None:  
            # create default params  
            params = ADMMParameters()  
-        # todo: consider keeping params as an object instead of copying

Note: you're not copying here, if the objects are a list they are taken via reference. Do you want to copy?

[Anton]

In [qiskit/optimization/algorithms/admm\_optimizer.py](https://urldefense.proofpoint.com/v2/url?u=https-3A__github.com_stefan-2Dwoerner_qiskit-2Daqua_pull_50-23discussion-5Fr396312251&d=DwMFaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=sqbsN76ZNhxb9WaLQsmrvHpMr-81A1A_aEDd-6W-z4w&m=XSEp6YB7H5KwkL344WgK8egaonOg7dRhK9gtVTWU4pY&s=HlBYlq-LMkAMHdc50AxigGQGfH7dBrrDYWCfrihNsEc&e=" \t "_blank):

>   
        # internal state where we'll keep intermediate solution  
-        # here, we just declare the class variable  
+        # here, we just declare the class variable, the variable is initialized in kept in the solve method.  
        self.\_state = None  
  
    def is\_compatible(self, problem: OptimizationProblem):

Add return type

⬇️ Suggested change  
-    def is\_compatible(self, problem: OptimizationProblem):  
+    def is\_compatible(self, problem: OptimizationProblem) -> Optional[str]:

[Anton]

In [qiskit/optimization/algorithms/admm\_optimizer.py](https://urldefense.proofpoint.com/v2/url?u=https-3A__github.com_stefan-2Dwoerner_qiskit-2Daqua_pull_50-23discussion-5Fr396313803&d=DwMFaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=sqbsN76ZNhxb9WaLQsmrvHpMr-81A1A_aEDd-6W-z4w&m=XSEp6YB7H5KwkL344WgK8egaonOg7dRhK9gtVTWU4pY&s=APF922e5AM2W_0penE5n_G-iBM4jeipZIIhbcoYy_g8&e=" \t "_blank):

> @@ -181,7 +208,6 @@ def is\_compatible(self, problem: OptimizationProblem):  
            # quadratic constraints are not supported  
            return "Quadratic constraints are not supported"  
  
-        # todo: verify other properties of the problem  
        return None  
  
    def solve(self, problem: OptimizationProblem):

Add return type

⬇️ Suggested change  
-    def solve(self, problem: OptimizationProblem):  
+    def solve(self, problem: OptimizationProblem) -> OptimizationResult:

[Anton]

In [qiskit/optimization/algorithms/admm\_optimizer.py](https://urldefense.proofpoint.com/v2/url?u=https-3A__github.com_stefan-2Dwoerner_qiskit-2Daqua_pull_50-23discussion-5Fr396315920&d=DwMFaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=sqbsN76ZNhxb9WaLQsmrvHpMr-81A1A_aEDd-6W-z4w&m=XSEp6YB7H5KwkL344WgK8egaonOg7dRhK9gtVTWU4pY&s=Pq2lJ5zQ0K0MU_EYwAjoRxoDrqV43dU0m5EHuQndVzE&e=" \t "_blank):

>   
class ADMMParameters:  
+    """Defines a set of parameters for ADMM optimizer.  
+    """  
+  
    def \_\_init\_\_(self, rho\_initial=10000, factor\_c=100000, beta=1000, max\_iter=10, tol=1.e-4, max\_time=1800,

What's the reason for setting the max\_time to 1800 seconds? Depending on problem size and the machine you're using this could more, couldn't it? I think a user would expect the default behaviour is that the algorithm runs as long as necessary, and as additional feature we can limit the runtime. We could change this to np.inf as default value or None and then check whether a maximum time is set.

[Claudio to accept the suggestion]

In [qiskit/optimization/algorithms/admm\_optimizer.py](https://urldefense.proofpoint.com/v2/url?u=https-3A__github.com_stefan-2Dwoerner_qiskit-2Daqua_pull_50-23discussion-5Fr396319219&d=DwMFaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=sqbsN76ZNhxb9WaLQsmrvHpMr-81A1A_aEDd-6W-z4w&m=XSEp6YB7H5KwkL344WgK8egaonOg7dRhK9gtVTWU4pY&s=RfUIGAZeBUS513hriR0PMTXbicxaVPfc39Exlz4j780&e=" \t "_blank):

>              # debug  
-            op1.write("op1.lp")  
+            # op1.write("op1.lp")

If we want to integrate debug facilities the users shouldn't need to go into the actual file and uncomment certain lines.  
A more elegant solution would be adding a debug argument in the initializer and then have

if debug:  
  op1.write('op1.lp')

or, whats commonly used, use a logger to write additional information.  
For this we need to define a top-level logger (at the top of your file)

import ...  
import logging  
  
logger = logging.getLogger(\_\_name\_\_)

and then we can use

logger.write('msg')

See [https://docs.python.org/3.7/library/logging.html](https://urldefense.proofpoint.com/v2/url?u=https-3A__docs.python.org_3.7_library_logging.html&d=DwMFaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=sqbsN76ZNhxb9WaLQsmrvHpMr-81A1A_aEDd-6W-z4w&m=XSEp6YB7H5KwkL344WgK8egaonOg7dRhK9gtVTWU4pY&s=5VZfzH4tQT0Acxf2jeGpe56j1PZHLt3Br0h1LKtqde4&e=) for more information.

[Anton. Claudio: yes it would be good to keep a debugging option, if that’s not too cumbersome. However I am not sure creating files such as .lp files is a common practice in a debug mode, we can discuss this.]

In [qiskit/optimization/algorithms/admm\_optimizer.py](https://urldefense.proofpoint.com/v2/url?u=https-3A__github.com_stefan-2Dwoerner_qiskit-2Daqua_pull_50-23discussion-5Fr396319393&d=DwMFaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=sqbsN76ZNhxb9WaLQsmrvHpMr-81A1A_aEDd-6W-z4w&m=XSEp6YB7H5KwkL344WgK8egaonOg7dRhK9gtVTWU4pY&s=Nvpfk-DH5EPcqjYt_TqCgQKIQNaZDxJyvpwmFlLQ3Gc&e=" \t "_blank):

>   
            self.\_state.x0 = self.update\_x0(op1)  
            # debug  
-            print("x0={}".format(self.\_state.x0))  
+            # print("x0={}".format(self.\_state.x0))

Use logger instead (also for all commented print statements you use below).

[Anton]

In [qiskit/optimization/algorithms/admm\_optimizer.py](https://urldefense.proofpoint.com/v2/url?u=https-3A__github.com_stefan-2Dwoerner_qiskit-2Daqua_pull_50-23discussion-5Fr396319949&d=DwMFaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=sqbsN76ZNhxb9WaLQsmrvHpMr-81A1A_aEDd-6W-z4w&m=XSEp6YB7H5KwkL344WgK8egaonOg7dRhK9gtVTWU4pY&s=RJU-zjfhnwQalP2J3-nSMw34RfLMABWDcZfrDs8Za74&e=" \t "_blank):

>   
            lambda\_mult = self.update\_lambda\_mult()  
  
-            cost\_iterate = self.get\_cost\_val()  
-  
-            cr = self.get\_cons\_res()  
-  
-            r, s = self.get\_sol\_res(it)  
-  
+            cost\_iterate = self.get\_objective\_value()  
+            cr = self.get\_constraint\_residual()

I think cr is an invalid name for pylint, consider changing this to constraint\_residual.

[Anton, please accept the suggestion]

In [qiskit/optimization/algorithms/admm\_optimizer.py](https://urldefense.proofpoint.com/v2/url?u=https-3A__github.com_stefan-2Dwoerner_qiskit-2Daqua_pull_50-23discussion-5Fr396320470&d=DwMFaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=sqbsN76ZNhxb9WaLQsmrvHpMr-81A1A_aEDd-6W-z4w&m=XSEp6YB7H5KwkL344WgK8egaonOg7dRhK9gtVTWU4pY&s=8Zsp_iqDIfIsFMQYFCFwa1XankS9pYa1wyOT6iGTnAI&e=" \t "_blank):

> @@ -291,11 +311,31 @@ def \_get\_variable\_indices(op: OptimizationProblem, var\_type: str) -> List[int]:  
  
        return indices  
  
-    def get\_q0(self):  
-        return self.\_get\_q(self.\_state.binary\_indices)  
-  
-    def get\_q1(self):  
-        return self.\_get\_q(self.\_state.continuous\_indices)  
+    def revert\_solution\_indexes(self, internal\_solution) -> np.ndarray:

This misses the type hint for internal\_solution and the docstring.

[Anton, type hint, Claudio docstring]

In [qiskit/optimization/algorithms/admm\_optimizer.py](https://urldefense.proofpoint.com/v2/url?u=https-3A__github.com_stefan-2Dwoerner_qiskit-2Daqua_pull_50-23discussion-5Fr396320700&d=DwMFaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=sqbsN76ZNhxb9WaLQsmrvHpMr-81A1A_aEDd-6W-z4w&m=XSEp6YB7H5KwkL344WgK8egaonOg7dRhK9gtVTWU4pY&s=5DvqGPnd9tZ9gUNrGNw1lLJx2_U_IHHqdGuaHwnoWAw&e=" \t "_blank):

> @@ -291,11 +311,31 @@ def \_get\_variable\_indices(op: OptimizationProblem, var\_type: str) -> List[int]:  
  
        return indices  
  
-    def get\_q0(self):  
-        return self.\_get\_q(self.\_state.binary\_indices)  
-  
-    def get\_q1(self):  
-        return self.\_get\_q(self.\_state.continuous\_indices)  
+    def revert\_solution\_indexes(self, internal\_solution) -> np.ndarray:  
+        (x0, u) = internal\_solution

Brackets are not needed, if you want you can just use

⬇️ Suggested change  
-        (x0, u) = internal\_solution  
+        x0, u = internal\_solution

[Claudio]

In [qiskit/optimization/algorithms/admm\_optimizer.py](https://urldefense.proofpoint.com/v2/url?u=https-3A__github.com_stefan-2Dwoerner_qiskit-2Daqua_pull_50-23discussion-5Fr396321335&d=DwMFaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=sqbsN76ZNhxb9WaLQsmrvHpMr-81A1A_aEDd-6W-z4w&m=XSEp6YB7H5KwkL344WgK8egaonOg7dRhK9gtVTWU4pY&s=5TXtTeMfif0g6QUDCnT8Hpi4U2ibuw5b61UHFsG6nOo&e=" \t "_blank):

> +        [solution.itemset(binary\_index, x0[i]) for i, binary\_index in enumerate(self.\_state.binary\_indices)]  
+        [solution.itemset(continuous\_index, u[i]) for i, continuous\_index in enumerate(self.\_state.continuous\_indices)]

This will raise an error in pylint afaik since the list is not assigned to anything. Use this workaround instead:

⬇️ Suggested change  
-        [solution.itemset(binary\_index, x0[i]) for i, binary\_index in enumerate(self.\_state.binary\_indices)]  
-        [solution.itemset(continuous\_index, u[i]) for i, continuous\_index in enumerate(self.\_state.continuous\_indices)]  
+        \_ = [solution.itemset(binary\_index, x0[i]) for i, binary\_index in enumerate(self.\_state.binary\_indices)]  
+        \_ = [solution.itemset(continuous\_index, u[i]) for i, continuous\_index in enumerate(self.\_state.continuous\_indices)]

[Anton]

In [qiskit/optimization/algorithms/admm\_optimizer.py](https://urldefense.proofpoint.com/v2/url?u=https-3A__github.com_stefan-2Dwoerner_qiskit-2Daqua_pull_50-23discussion-5Fr396321606&d=DwMFaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=sqbsN76ZNhxb9WaLQsmrvHpMr-81A1A_aEDd-6W-z4w&m=XSEp6YB7H5KwkL344WgK8egaonOg7dRhK9gtVTWU4pY&s=pB02ujqFzRaFHMkOgAK6O75sVgkrKdgdLrTO5QTeUXc&e=" \t "_blank):

> @@ -291,11 +311,31 @@ def \_get\_variable\_indices(op: OptimizationProblem, var\_type: str) -> List[int]:  
  
        return indices  
  
-    def get\_q0(self):  
-        return self.\_get\_q(self.\_state.binary\_indices)  
-  
-    def get\_q1(self):  
-        return self.\_get\_q(self.\_state.continuous\_indices)  
+    def revert\_solution\_indexes(self, internal\_solution) -> np.ndarray:  
+        (x0, u) = internal\_solution  
+        solution = np.zeros(len(self.\_state.binary\_indices) + len(self.\_state.continuous\_indices))  
+        # restore solution at the original index location  
+        [solution.itemset(binary\_index, x0[i]) for i, binary\_index in enumerate(self.\_state.binary\_indices)]  
+        [solution.itemset(continuous\_index, u[i]) for i, continuous\_index in enumerate(self.\_state.continuous\_indices)]  
+        return solution  
+  
+    def convert\_problem\_representation(self) -> None:

Misses the docstring.

[Claudio]

In [qiskit/optimization/algorithms/admm\_optimizer.py](https://urldefense.proofpoint.com/v2/url?u=https-3A__github.com_stefan-2Dwoerner_qiskit-2Daqua_pull_50-23discussion-5Fr396322219&d=DwMFaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=sqbsN76ZNhxb9WaLQsmrvHpMr-81A1A_aEDd-6W-z4w&m=XSEp6YB7H5KwkL344WgK8egaonOg7dRhK9gtVTWU4pY&s=mlu2Wsia9Wy3g9lYUdrUO7Rx-GV_9pZiR4VLLTmbHd0&e=" \t "_blank):

> +        [solution.itemset(continuous\_index, u[i]) for i, continuous\_index in enumerate(self.\_state.continuous\_indices)]  
+        return solution  
+  
+    def convert\_problem\_representation(self) -> None:  
+        # objective  
+        self.get\_q0()  
+        self.get\_c0()  
+        self.get\_q1()  
+        self.get\_c1()  
+        # constraints  
+        self.get\_a0\_b0()  
+        self.get\_a1\_b1()  
+        self.get\_a2\_a3\_b2()  
+        self.get\_a4\_b3()  
+  
+    def get\_q0(self) -> None:

Misses the docstring. Also the name q0 is not very descriptive, but this is the same remark I had above about the cache. I think it would be really helpful if you could add a reference paper where all these variable names are used.

[Claudio]

In [qiskit/optimization/algorithms/admm\_optimizer.py](https://urldefense.proofpoint.com/v2/url?u=https-3A__github.com_stefan-2Dwoerner_qiskit-2Daqua_pull_50-23discussion-5Fr396322346&d=DwMFaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=sqbsN76ZNhxb9WaLQsmrvHpMr-81A1A_aEDd-6W-z4w&m=XSEp6YB7H5KwkL344WgK8egaonOg7dRhK9gtVTWU4pY&s=wja9ZNDOJEzJGM0GC-wniH3x2ywXp-MK6mwTSFerU78&e=" \t "_blank):

> +    def convert\_problem\_representation(self) -> None:  
+        # objective  
+        self.get\_q0()  
+        self.get\_c0()  
+        self.get\_q1()  
+        self.get\_c1()  
+        # constraints  
+        self.get\_a0\_b0()  
+        self.get\_a1\_b1()  
+        self.get\_a2\_a3\_b2()  
+        self.get\_a4\_b3()  
+  
+    def get\_q0(self) -> None:  
+        self.\_state.q0 = self.\_get\_q(self.\_state.binary\_indices)  
+  
+    def get\_q1(self) -> None:

Missing docstring

[Claudio]

In [qiskit/optimization/algorithms/admm\_optimizer.py](https://urldefense.proofpoint.com/v2/url?u=https-3A__github.com_stefan-2Dwoerner_qiskit-2Daqua_pull_50-23discussion-5Fr396322628&d=DwMFaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=sqbsN76ZNhxb9WaLQsmrvHpMr-81A1A_aEDd-6W-z4w&m=XSEp6YB7H5KwkL344WgK8egaonOg7dRhK9gtVTWU4pY&s=Yq5gTr5v06DsAW66PZyAbt4HEWn2zxdZL6uiCVjNl8k&e=" \t "_blank):

> @@ -315,11 +355,11 @@ def \_get\_c(self, variable\_indices: List[int]) -> np.ndarray:  
        c = c \* self.\_state.sense

You could use in-place multiplication here

⬇️ Suggested change  
-        c = c \* self.\_state.sense  
+        c \*= self.\_state.sense

[Claudio]

In [qiskit/optimization/algorithms/admm\_optimizer.py](https://urldefense.proofpoint.com/v2/url?u=https-3A__github.com_stefan-2Dwoerner_qiskit-2Daqua_pull_50-23discussion-5Fr396322734&d=DwMFaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=sqbsN76ZNhxb9WaLQsmrvHpMr-81A1A_aEDd-6W-z4w&m=XSEp6YB7H5KwkL344WgK8egaonOg7dRhK9gtVTWU4pY&s=8U851I3bAvPP2KoxebpVGtgud1nUUGr1M1o8Yjq3TmM&e=" \t "_blank):

> @@ -315,11 +355,11 @@ def \_get\_c(self, variable\_indices: List[int]) -> np.ndarray:  
        c = c \* self.\_state.sense  
        return c  
  
-    def get\_c0(self):  
-        return self.\_get\_c(self.\_state.binary\_indices)  
+    def get\_c0(self) -> None:

* Missing docstring
* The name get\_ is not very accurate, since this is not returning anything. It would be better to use set\_c0 instead.

[Claudio for the docstring. Anton, do you agree with refactoring to set? The object has not been created before. I am not an expert of getters and setters here :D]

In [qiskit/optimization/algorithms/admm\_optimizer.py](https://urldefense.proofpoint.com/v2/url?u=https-3A__github.com_stefan-2Dwoerner_qiskit-2Daqua_pull_50-23discussion-5Fr396322807&d=DwMFaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=sqbsN76ZNhxb9WaLQsmrvHpMr-81A1A_aEDd-6W-z4w&m=XSEp6YB7H5KwkL344WgK8egaonOg7dRhK9gtVTWU4pY&s=OkVyz_PviUOdfRGHEc4uf23bjb7kYToJ8MRijKrCg5A&e=" \t "_blank):

>   
-    def get\_c1(self):  
-        return self.\_get\_c(self.\_state.continuous\_indices)  
+    def get\_c1(self) -> None:

* Missing docstring
* Use set\_ instead of get\_  
    
  [Claudio for the docstring. Anton, do you agree with refactoring to set? The object has not been created before. I am not an expert of getters and setters here :D]

In [qiskit/optimization/algorithms/admm\_optimizer.py](https://urldefense.proofpoint.com/v2/url?u=https-3A__github.com_stefan-2Dwoerner_qiskit-2Daqua_pull_50-23discussion-5Fr396323915&d=DwMFaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=sqbsN76ZNhxb9WaLQsmrvHpMr-81A1A_aEDd-6W-z4w&m=XSEp6YB7H5KwkL344WgK8egaonOg7dRhK9gtVTWU4pY&s=CBKrBRrn0E0ZrJIkhhuZEhu-07ZTxT0h_o0jNIxyrls&e=" \t "_blank):

> +        # objective  
+        self.get\_q0()  
+        self.get\_c0()  
+        self.get\_q1()  
+        self.get\_c1()  
+        # constraints  
+        self.get\_a0\_b0()  
+        self.get\_a1\_b1()  
+        self.get\_a2\_a3\_b2()  
+        self.get\_a4\_b3()

This is a bit confusing: all these methods are named get\_something but they aren't returning anything? Consider using set\_something instead since they are setting the internal values. 🙂

[Anton]

In [qiskit/optimization/algorithms/admm\_optimizer.py](https://urldefense.proofpoint.com/v2/url?u=https-3A__github.com_stefan-2Dwoerner_qiskit-2Daqua_pull_50-23discussion-5Fr396324090&d=DwMFaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=sqbsN76ZNhxb9WaLQsmrvHpMr-81A1A_aEDd-6W-z4w&m=XSEp6YB7H5KwkL344WgK8egaonOg7dRhK9gtVTWU4pY&s=hi9fNgkGOhw2eOeZaS6YoddXauQKpq_MLIzTllMGhno&e=" \t "_blank):

>          # if we don't have such constraints, return just dummy arrays  
        if len(matrix) != 0:  
            return np.array(matrix), np.array(vector)  
        else:  
            return np.array([0] \* size).reshape((1, -1)), np.zeros(shape=(1,))  
  
-    def get\_a0\_b0(self) -> (np.ndarray, np.ndarray):  
+    def get\_a0\_b0(self) -> None:

* Missing docstring
* Use set\_ instead of get\_

[Claudio docstring, Anton set get]

In [qiskit/optimization/algorithms/admm\_optimizer.py](https://urldefense.proofpoint.com/v2/url?u=https-3A__github.com_stefan-2Dwoerner_qiskit-2Daqua_pull_50-23discussion-5Fr396324380&d=DwMFaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=sqbsN76ZNhxb9WaLQsmrvHpMr-81A1A_aEDd-6W-z4w&m=XSEp6YB7H5KwkL344WgK8egaonOg7dRhK9gtVTWU4pY&s=jR1ciU9VEH_gW-j0jMvqo5zdfPItHhgUfuSDeqg53tk&e=" \t "_blank):

> @@ -382,16 +423,15 @@ def \_get\_inequality\_matrix\_and\_vector(self, variable\_indices: List[int]) -> (Lis  
  
        return matrix, vector  
  
-    def get\_a1\_b1(self) -> (np.ndarray, np.ndarray):  
+    def get\_a1\_b1(self) -> None:

* Missing docstring
* Use set\_ instead of get\_  
    
    
  [Claudio docstring, Anton set get]

In [qiskit/optimization/algorithms/admm\_optimizer.py](https://urldefense.proofpoint.com/v2/url?u=https-3A__github.com_stefan-2Dwoerner_qiskit-2Daqua_pull_50-23discussion-5Fr396324443&d=DwMFaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=sqbsN76ZNhxb9WaLQsmrvHpMr-81A1A_aEDd-6W-z4w&m=XSEp6YB7H5KwkL344WgK8egaonOg7dRhK9gtVTWU4pY&s=a0Z6ykiT6CBwG6tbgo_zRutBEJDb_F3JkkkuL5cwpso&e=" \t "_blank):

>   
-    def get\_a4\_b3(self) -> (np.ndarray, np.ndarray):  
+    def get\_a4\_b3(self) -> None:

* Missing docstring
* Use set\_ instead of get\_  
    
  [Claudio docstring, Anton set get]

In [qiskit/optimization/algorithms/admm\_optimizer.py](https://urldefense.proofpoint.com/v2/url?u=https-3A__github.com_stefan-2Dwoerner_qiskit-2Daqua_pull_50-23discussion-5Fr396324515&d=DwMFaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=sqbsN76ZNhxb9WaLQsmrvHpMr-81A1A_aEDd-6W-z4w&m=XSEp6YB7H5KwkL344WgK8egaonOg7dRhK9gtVTWU4pY&s=ZXHITmDKxweb_z7tcctNOx_P7fHDhkGMBItPetCmlRA&e=" \t "_blank):

>   
-        return self.\_create\_ndarrays(matrix, vector, len(self.\_state.continuous\_indices))  
-  
-    def get\_a2\_a3\_b2(self) -> (np.ndarray, np.ndarray, np.ndarray):  
+    def get\_a2\_a3\_b2(self) -> None:

* Missing docstring
* Use set\_ instead of get\_  
    
    
  [Claudio docstring, Anton set get]

In [qiskit/optimization/algorithms/admm\_optimizer.py](https://urldefense.proofpoint.com/v2/url?u=https-3A__github.com_stefan-2Dwoerner_qiskit-2Daqua_pull_50-23discussion-5Fr396327009&d=DwMFaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=sqbsN76ZNhxb9WaLQsmrvHpMr-81A1A_aEDd-6W-z4w&m=XSEp6YB7H5KwkL344WgK8egaonOg7dRhK9gtVTWU4pY&s=7HK33dm5tqZHTRSH7RoY9BNyRQ_qw1wTWSepqbg1Oyk&e=" \t "_blank):

> @@ -547,43 +586,43 @@ def \_create\_step3\_problem(self):  
  
    # when a plain list() call is used a numpy type of values makes cplex to fail when cplex.write() is called.  
    # for debug only, list() should be used instead  
-    def \_to\_list(self, values):  
+    @staticmethod  
+    def \_to\_list(values):

Is this method really needed? You can change this in the code from

list\_as\_float = ADMMOptimizer.\_to\_list(other\_list)

to

list\_as\_float = [float(el) for el in other\_list]

This is even shorter and it is immediately clear what's going on. Otherwise, if people read the code, they first have to go see what ADMMOptimizer.\_to\_list does.

[Anton]

In [qiskit/optimization/algorithms/admm\_optimizer.py](https://urldefense.proofpoint.com/v2/url?u=https-3A__github.com_stefan-2Dwoerner_qiskit-2Daqua_pull_50-23discussion-5Fr396327454&d=DwMFaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=sqbsN76ZNhxb9WaLQsmrvHpMr-81A1A_aEDd-6W-z4w&m=XSEp6YB7H5KwkL344WgK8egaonOg7dRhK9gtVTWU4pY&s=_yy7q6uqlWjnY4JDmshrSa3BuazS8aYMLJ5p7ZkhLRk&e=" \t "_blank):

> @@ -547,43 +586,43 @@ def \_create\_step3\_problem(self):  
  
    # when a plain list() call is used a numpy type of values makes cplex to fail when cplex.write() is called.  
    # for debug only, list() should be used instead  
-    def \_to\_list(self, values):  
+    @staticmethod  
+    def \_to\_list(values):  
        out\_list = []  
        for el in values:  
            out\_list.append(float(el))  
        return out\_list  
  
    def update\_x0(self, op1: OptimizationProblem) -> np.ndarray:

Missing docstring

[Claudio]

In [qiskit/optimization/algorithms/admm\_optimizer.py](https://urldefense.proofpoint.com/v2/url?u=https-3A__github.com_stefan-2Dwoerner_qiskit-2Daqua_pull_50-23discussion-5Fr396327535&d=DwMFaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=sqbsN76ZNhxb9WaLQsmrvHpMr-81A1A_aEDd-6W-z4w&m=XSEp6YB7H5KwkL344WgK8egaonOg7dRhK9gtVTWU4pY&s=DEx7LExBSgudG5ow7ZTdWvoh6v5KLvlztSNNL71W6sM&e=" \t "_blank):

>          out\_list = []  
        for el in values:  
            out\_list.append(float(el))  
        return out\_list  
  
    def update\_x0(self, op1: OptimizationProblem) -> np.ndarray:  
-        # TODO: Check output type of qubo\_solver.solve(op1).x  
-        return np.asarray(self.\_qubo\_solver.solve(op1).x)  
+        return np.asarray(self.\_qubo\_optimizer.solve(op1).x)  
  
    def update\_x1(self, op2: OptimizationProblem) -> (np.ndarray, np.ndarray):

Missing docstring  
[Claudio]

In [qiskit/optimization/algorithms/admm\_optimizer.py](https://urldefense.proofpoint.com/v2/url?u=https-3A__github.com_stefan-2Dwoerner_qiskit-2Daqua_pull_50-23discussion-5Fr396327933&d=DwMFaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=sqbsN76ZNhxb9WaLQsmrvHpMr-81A1A_aEDd-6W-z4w&m=XSEp6YB7H5KwkL344WgK8egaonOg7dRhK9gtVTWU4pY&s=OTVaWZtmpbsBnS_HN8HccxD0U83Ftgs0UcXGLibmm_c&e=" \t "_blank):

>   
    def update\_x1(self, op2: OptimizationProblem) -> (np.ndarray, np.ndarray):  
-        vars\_op2 = self.\_continuous\_solver.solve(op2).x  
-        # TODO: Check output type  
+        vars\_op2 = self.\_continuous\_optimizer.solve(op2).x  
        u = np.asarray(vars\_op2[:len(self.\_state.continuous\_indices)])  
        z = np.asarray(vars\_op2[len(self.\_state.continuous\_indices):])  
        return u, z  
  
    def update\_y(self, op3):

Missing docstring  
  
[Claudio]

In [qiskit/optimization/algorithms/admm\_optimizer.py](https://urldefense.proofpoint.com/v2/url?u=https-3A__github.com_stefan-2Dwoerner_qiskit-2Daqua_pull_50-23discussion-5Fr396328499&d=DwMFaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=sqbsN76ZNhxb9WaLQsmrvHpMr-81A1A_aEDd-6W-z4w&m=XSEp6YB7H5KwkL344WgK8egaonOg7dRhK9gtVTWU4pY&s=HYRqDbaZntpAzADZDIiI8f0arSYCBcO3ijDv2K3jcIQ&e=" \t "_blank):

> +    def get\_best\_merit\_solution(self):  
        """

* Misses return type
* Misses single-line summary in the first line of the docstring  
    
  [Claudio]

In [qiskit/optimization/algorithms/admm\_optimizer.py](https://urldefense.proofpoint.com/v2/url?u=https-3A__github.com_stefan-2Dwoerner_qiskit-2Daqua_pull_50-23discussion-5Fr396328826&d=DwMFaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=sqbsN76ZNhxb9WaLQsmrvHpMr-81A1A_aEDd-6W-z4w&m=XSEp6YB7H5KwkL344WgK8egaonOg7dRhK9gtVTWU4pY&s=kKPepkoK2HAW6AgwDj1_JWYX_BlOSMp_zON_moaYq_4&e=" \t "_blank):

>          sol = [x0, u]  
-        sol\_val = self.\_state.cost\_iterates[it\_min\_merits]  
+        sol\_val = self.\_state.cost\_iterates[it\_best\_merits]  
        return sol, sol\_val  
  
    def update\_lambda\_mult(self):

Misses docstring and return type.  
Also the next function update\_rho misses type hints (but for some reason I cannot leave a comment in-line there).  
  
[Claudio]

In [qiskit/optimization/algorithms/admm\_optimizer.py](https://urldefense.proofpoint.com/v2/url?u=https-3A__github.com_stefan-2Dwoerner_qiskit-2Daqua_pull_50-23discussion-5Fr396329670&d=DwMFaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=sqbsN76ZNhxb9WaLQsmrvHpMr-81A1A_aEDd-6W-z4w&m=XSEp6YB7H5KwkL344WgK8egaonOg7dRhK9gtVTWU4pY&s=9aAtwRuFOD0j3p4YfRVeM6f-gLV95fE4w1zlvvofZsI&e=" \t "_blank):

> @@ -608,137 +647,76 @@ def update\_rho(self, r, s):  
            elif s > self.\_mu\_res \* r:  
                self.\_state.rho = self.\_tau\_decr \* self.\_state.rho  
  
-    def get\_cons\_res(self):  
+    def get\_constraint\_residual(self):

Missing return type and single-line summary  
  
[Claudio]

In [qiskit/optimization/algorithms/admm\_optimizer.py](https://urldefense.proofpoint.com/v2/url?u=https-3A__github.com_stefan-2Dwoerner_qiskit-2Daqua_pull_50-23discussion-5Fr396329801&d=DwMFaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=sqbsN76ZNhxb9WaLQsmrvHpMr-81A1A_aEDd-6W-z4w&m=XSEp6YB7H5KwkL344WgK8egaonOg7dRhK9gtVTWU4pY&s=U_t-yN73dwDOFz3MMTCxHA542gvlt34HZo4BnGBWqDg&e=" \t "_blank):

>          cr2 = sum(max(val, 0) for val in eq2)  
  
-        return cr0+cr1+cr2  
+        return cr0 + cr1 + cr2  
  
    def get\_merit(self, cost\_iterate, cr):

* Misses type hints
* cr is an invalid name for pylint (use constraint\_residual instead?)  
    
  [Anton]

In [qiskit/optimization/algorithms/admm\_optimizer.py](https://urldefense.proofpoint.com/v2/url?u=https-3A__github.com_stefan-2Dwoerner_qiskit-2Daqua_pull_50-23discussion-5Fr396329954&d=DwMFaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=sqbsN76ZNhxb9WaLQsmrvHpMr-81A1A_aEDd-6W-z4w&m=XSEp6YB7H5KwkL344WgK8egaonOg7dRhK9gtVTWU4pY&s=yFGfS9kFAAJZn6L2FauJqtgiw_7rdC9gTQQDQ4YmDt0&e=" \t "_blank):

>          """  
        Compute merit value associated with the current iterate  
  
⬇️ Suggested change  
-        """  
-        Compute merit value associated with the current iterate  
+        """Compute merit value associated with the current iterate.

[Anton]

In [qiskit/optimization/algorithms/admm\_optimizer.py](https://urldefense.proofpoint.com/v2/url?u=https-3A__github.com_stefan-2Dwoerner_qiskit-2Daqua_pull_50-23discussion-5Fr396330397&d=DwMFaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=sqbsN76ZNhxb9WaLQsmrvHpMr-81A1A_aEDd-6W-z4w&m=XSEp6YB7H5KwkL344WgK8egaonOg7dRhK9gtVTWU4pY&s=FR5y0LAJzbbxREfu6VOgQRMZxvRg3__gk26JFIbbYlk&e=" \t "_blank):

> +            cost\_iterate: cost at the certain iteration  
+            cr: value of violation of the constraints  
+  
+        Returns:  
+            merit value as a float

For consistent style, use full sentences

⬇️ Suggested change  
-            cost\_iterate: cost at the certain iteration  
-            cr: value of violation of the constraints  
-  
-        Returns:  
-            merit value as a float  
+            cost\_iterate: Cost at the certain iteration.  
+            cr: Value of violation of the constraints.  
+  
+        Returns:  
+            Merit value as a float.

[Claudio]

In [qiskit/optimization/algorithms/admm\_optimizer.py](https://urldefense.proofpoint.com/v2/url?u=https-3A__github.com_stefan-2Dwoerner_qiskit-2Daqua_pull_50-23discussion-5Fr396331022&d=DwMFaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=sqbsN76ZNhxb9WaLQsmrvHpMr-81A1A_aEDd-6W-z4w&m=XSEp6YB7H5KwkL344WgK8egaonOg7dRhK9gtVTWU4pY&s=sHFs5noinkXYMWVsJykx1ryo2N4zVmKpL04VAsgXrmI&e=" \t "_blank):

>          """  
        return cost\_iterate + self.\_mu \* cr  
  
-    def get\_cost\_val(self):  
+    def get\_objective\_value(self):

Misses return type

[Claudio]

In [qiskit/optimization/algorithms/admm\_optimizer.py](https://urldefense.proofpoint.com/v2/url?u=https-3A__github.com_stefan-2Dwoerner_qiskit-2Daqua_pull_50-23discussion-5Fr396331137&d=DwMFaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=sqbsN76ZNhxb9WaLQsmrvHpMr-81A1A_aEDd-6W-z4w&m=XSEp6YB7H5KwkL344WgK8egaonOg7dRhK9gtVTWU4pY&s=ycnM8jCIyqPDYR0wTmlOombv4nWyrF7ob2hBC6X9JcA&e=" \t "_blank):

>          """  
        Computes the value of the objective function.  
  
⬇️ Suggested change  
-        """  
-        Computes the value of the objective function.  
+        """Computes the value of the objective function.

[Anton]

In [qiskit/optimization/algorithms/admm\_optimizer.py](https://urldefense.proofpoint.com/v2/url?u=https-3A__github.com_stefan-2Dwoerner_qiskit-2Daqua_pull_50-23discussion-5Fr396331625&d=DwMFaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=sqbsN76ZNhxb9WaLQsmrvHpMr-81A1A_aEDd-6W-z4w&m=XSEp6YB7H5KwkL344WgK8egaonOg7dRhK9gtVTWU4pY&s=coQGqqs9c-61V6f3cBcjalssFy1Nx3PkHJwtQHvztHc&e=" \t "_blank):

>   
        return obj\_val  
  
-    def get\_sol\_res(self, it):  
+    def get\_solution\_residuals(self, it):

* Misses return type
* it is an invalid name for pylint (use iteration instead)

[Claudio]

In [qiskit/optimization/algorithms/admm\_optimizer.py](https://urldefense.proofpoint.com/v2/url?u=https-3A__github.com_stefan-2Dwoerner_qiskit-2Daqua_pull_50-23discussion-5Fr396331743&d=DwMFaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=sqbsN76ZNhxb9WaLQsmrvHpMr-81A1A_aEDd-6W-z4w&m=XSEp6YB7H5KwkL344WgK8egaonOg7dRhK9gtVTWU4pY&s=Rksjl6S3-HIoxfoeDWZhQ3YWh34jsDQ8q_6qzhdZp1U&e=" \t "_blank):

>          """  
        Compute primal and dual residual.  
  
⬇️ Suggested change  
-        """  
-        Compute primal and dual residual.  
+        """Compute primal and dual residual.  
  
  
[Claudio]

In [qiskit/optimization/algorithms/admm\_optimizer.py](https://urldefense.proofpoint.com/v2/url?u=https-3A__github.com_stefan-2Dwoerner_qiskit-2Daqua_pull_50-23discussion-5Fr396332280&d=DwMFaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=sqbsN76ZNhxb9WaLQsmrvHpMr-81A1A_aEDd-6W-z4w&m=XSEp6YB7H5KwkL344WgK8egaonOg7dRhK9gtVTWU4pY&s=z3viTtRR5fT4ERRpwwR4qpsR6GtgQihruA_oQtN6Yts&e=" \t "_blank):

>          """  
        Compute primal and dual residual.  
  
        Args:  
-            it:  
+            it: iteration number  
+  
+        Returns:  
+            r, s as primary and dual residuals  
        """  
        elements = self.\_state.x0 - self.\_state.z - self.\_state.y  
        # debug  
        # elements = np.asarray([x0[i] - z[i] + y[i] for i in self.range\_x0\_vars])  
        r = pow(sum(e \*\* 2 for e in elements), 0.5)

You could just use np.linalg.norm(elements) instead (or add a comment that r is the norm)  
  
[Claudio]

In [test/optimization/test\_admm\_miskp.py](https://urldefense.proofpoint.com/v2/url?u=https-3A__github.com_stefan-2Dwoerner_qiskit-2Daqua_pull_50-23discussion-5Fr396332557&d=DwMFaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=sqbsN76ZNhxb9WaLQsmrvHpMr-81A1A_aEDd-6W-z4w&m=XSEp6YB7H5KwkL344WgK8egaonOg7dRhK9gtVTWU4pY&s=_-MmT1aEHG1JJwJibDUhRruH3LwtX6LU8fMLWCX3W8c&e=):

> @@ -1,199 +1,95 @@  
+# -\*-coding: utf-8 -\*-  
+# This code is part of Qiskit.  
+#  
+# (C) Copyright IBM 2000.  
  
⬇️ Suggested change  
-# (C) Copyright IBM 2000.  
+# (C) Copyright IBM 2020.  
  
[Claudio]

In [test/optimization/test\_admm\_miskp.py](https://urldefense.proofpoint.com/v2/url?u=https-3A__github.com_stefan-2Dwoerner_qiskit-2Daqua_pull_50-23discussion-5Fr396333025&d=DwMFaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=sqbsN76ZNhxb9WaLQsmrvHpMr-81A1A_aEDd-6W-z4w&m=XSEp6YB7H5KwkL344WgK8egaonOg7dRhK9gtVTWU4pY&s=ZqlAvXnGCZAlWYBhL1y9qarZeyMnncAHte8OI-vwoBc&e=):

> +    def setUp(self):  
+        super().setUp()

If you're not doing anything special in setUp you can leave it out

⬇️ Suggested change  
-    def setUp(self):  
-        super().setUp()

[Anton]

In [test/optimization/test\_admm\_miskp.py](https://urldefense.proofpoint.com/v2/url?u=https-3A__github.com_stefan-2Dwoerner_qiskit-2Daqua_pull_50-23discussion-5Fr396333565&d=DwMFaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=sqbsN76ZNhxb9WaLQsmrvHpMr-81A1A_aEDd-6W-z4w&m=XSEp6YB7H5KwkL344WgK8egaonOg7dRhK9gtVTWU4pY&s=hFwQg62b5nE23A3RaCcVH8RlI0zWxqLbDMeH2YpCddk&e=):

> +        K, T, P, S, D, C = self.get\_problem\_params()  
+        miskp = Miskp(K, T, P, S, D, C)

These names will raise an error in pylint, but you can just do this

⬇️ Suggested change  
-        K, T, P, S, D, C = self.get\_problem\_params()  
-        miskp = Miskp(K, T, P, S, D, C)  
+        miskp = Miskp(\*self.get\_problem\_params())

which uses cool python syntax 😉

[Anton]

In [test/optimization/test\_admm\_miskp.py](https://urldefense.proofpoint.com/v2/url?u=https-3A__github.com_stefan-2Dwoerner_qiskit-2Daqua_pull_50-23discussion-5Fr396334181&d=DwMFaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=sqbsN76ZNhxb9WaLQsmrvHpMr-81A1A_aEDd-6W-z4w&m=XSEp6YB7H5KwkL344WgK8egaonOg7dRhK9gtVTWU4pY&s=gS1LS2rqa8aKS3oDHySxYJ2cDvrwuztOxOl5A5CQiHE&e=):

>   
+    def test\_admm\_optimizer\_miskp\_eigen(self):  
+        """ ADMM Optimizer Test based on Mixed-Integer Setup Knapsack Problem using NumPy eigen optimizer"""  
+        K, T, P, S, D, C = self.get\_problem\_params()  
+        miskp = Miskp(K, T, P, S, D, C)  
+        op: OptimizationProblem = miskp.create\_problem()

There should be a test statement here, even if it just tests just some trivial property of op.  
  
[Anton]

In [test/optimization/test\_admm\_miskp.py](https://urldefense.proofpoint.com/v2/url?u=https-3A__github.com_stefan-2Dwoerner_qiskit-2Daqua_pull_50-23discussion-5Fr396334727&d=DwMFaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=sqbsN76ZNhxb9WaLQsmrvHpMr-81A1A_aEDd-6W-z4w&m=XSEp6YB7H5KwkL344WgK8egaonOg7dRhK9gtVTWU4pY&s=RrMUuQHtuhgJG0PDOCR0UK5YU14CnPo-t96pypA1fGg&e=):

> +        print("results")  
+        print("x={}".format(solution.x))  
+        print("fval={}".format(solution.fval))

Don't use default prints, either use a logger or remove them entirely. Otherwise the make test output will be crowded with random print statements.

⬇️ Suggested change  
-        print("results")  
-        print("x={}".format(solution.x))  
-        print("fval={}".format(solution.fval))

[Anton]

In [test/optimization/test\_admm\_miskp.py](https://urldefense.proofpoint.com/v2/url?u=https-3A__github.com_stefan-2Dwoerner_qiskit-2Daqua_pull_50-23discussion-5Fr396335515&d=DwMFaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=sqbsN76ZNhxb9WaLQsmrvHpMr-81A1A_aEDd-6W-z4w&m=XSEp6YB7H5KwkL344WgK8egaonOg7dRhK9gtVTWU4pY&s=2WIGPuOPxla2osXRUoTGnO_ebDF9A-IIt642auGZeY8&e=):

> +        print("fval={}".format(solution.fval))  
+  
+        correct\_solution = [0.009127, 0.009127, 0.009127, 0.009127, 0.009127, 0.009127, 0.009127, 0.009127,  
+                            0.009127, 0.009127, 0.006151, 0.006151, 0.006151, 0.006151, 0.006151, 0.006151,  
+                            0.006151, 0.006151, 0.006151, 0.006151, 0.,       0.]  
+        correct\_objective = -1.2113693  
+  
+        np.testing.assert\_almost\_equal(correct\_solution, solution.x, 3)  
+        np.testing.assert\_almost\_equal(solution.fval, correct\_objective, 3)  
+  
+    def get\_problem\_params(self):  
+        """  
+        Fills in parameters for a Mixed Integer Setup Knapsack Problem (MISKP) instance.  
+        """  
+        #  
+

These names will be invalid, could you give them proper names? Or, since this is just a test file, I think it's okay to add a global #pylint: disable=invalid-name at the top of the file if you'd prefer that.

[Anton]

In [test/optimization/test\_admm\_miskp.py](https://urldefense.proofpoint.com/v2/url?u=https-3A__github.com_stefan-2Dwoerner_qiskit-2Daqua_pull_50-23discussion-5Fr396335771&d=DwMFaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=sqbsN76ZNhxb9WaLQsmrvHpMr-81A1A_aEDd-6W-z4w&m=XSEp6YB7H5KwkL344WgK8egaonOg7dRhK9gtVTWU4pY&s=rS0O55ekDBzfcW09jtjEP5vNl-6Ut8BmRpktguTDlxc&e=):

>   
-    def \_\_init\_\_(self, K, T, P, S, D: np.ndarray, C: np.ndarray, verbose=False, relativeGap=0.0,  
-                 pairwise\_incomp=0, multiple\_choice=0):  
+  
+class Miskp:  
+    def \_\_init\_\_(self, K, T, P, S, D: np.ndarray, C: np.ndarray, pairwise\_incomp=0, multiple\_choice=0):

If you use type hints for some args I think you should use them for all 🙂

[Anton]—  
You are receiving this because you were mentioned.  
Reply to this email directly, [view it on GitHub](https://urldefense.proofpoint.com/v2/url?u=https-3A__github.com_stefan-2Dwoerner_qiskit-2Daqua_pull_50-23pullrequestreview-2D379229078&d=DwMFaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=sqbsN76ZNhxb9WaLQsmrvHpMr-81A1A_aEDd-6W-z4w&m=XSEp6YB7H5KwkL344WgK8egaonOg7dRhK9gtVTWU4pY&s=X_1FxTjwNXaJAJRA9bnMEuVJ4pK9rXGAgrCE-Dg3Dvs&e=), or [unsubscribe](https://urldefense.proofpoint.com/v2/url?u=https-3A__github.com_notifications_unsubscribe-2Dauth_AO3SJZQLXK3663V2GIBOJ3DRI4ZXVANCNFSM4LQU2CMQ&d=DwMFaQ&c=jf_iaSHvJObTbx-siA1ZOg&r=sqbsN76ZNhxb9WaLQsmrvHpMr-81A1A_aEDd-6W-z4w&m=XSEp6YB7H5KwkL344WgK8egaonOg7dRhK9gtVTWU4pY&s=GmEGIdLK69zPu_muLOpzO4JwBgOlTjigR310wYOj_pE&e=).