# Chapter 17

## Motif-Driven Design of Protein–Protein Interfaces

Daniel-Adriano Silva, Bruno E. Correia, and Erik Procko

### Abstract

Protein–protein interfaces regulate many critical processes for cellular function. The ability to accurately control and regulate these molecular interactions is of major interest for biomedical and synthetic biology applications, as well as to address fundamental biological questions. In recent years, computational protein design has emerged as a tool for designing novel protein–protein interactions with functional relevance. Although attractive, these computational tools carry a steep learning curve. In order to make some of these methods more accessible, we present detailed descriptions and examples of ROSETTA computational protocols for the design of functional protein binders using seeded protein interface design. In these protocols, a motif of known structure that interacts with the target site is grafted into a scaffold protein, followed by design of the surrounding interaction surface.

**Key words** Computational protein design, Protein–protein interaction, ROSETTA, Motif grafting, Interface design

## 1 Introduction

Computational design of protein–protein interactions has steadily progressed in recent years, including the creation of inhibitors that block enzymatic sites [1], small proteins that prevent viral entry [2], and antitumor agents that sequester oncogenic factors [3]. The ability to design in silico new functional binding proteins from minimal starting components opens tremendous possibilities for engineering innovative therapeutics and may eventually challenge antibody technology as the premiere method for generating protein-based drugs. However, designing a truly de novo protein–protein interface is a challenging problem that remains largely unsolved. This is due to several factors, most importantly the inaccuracies in energy functions used to evaluate protein designs and the intrinsic difficulties in efficiently sampling docked protein configurations that allow the design of side chains for favorable interactions. Therefore, to overcome these limitations, protein designers often use a "seeded interface design" approach, in which a small

Barry L. Stoddard (ed.), *Computational Design of Ligand Binding Proteins*, Methods in Molecular Biology, vol. 1414,
DOI 10.1007/978-1-4939-3569-7_17, © Springer Science+Business Media New York 2016

motif of known structure that binds to the target site is used to initiate the design process. This motif is then grafted (i.e., embedded) into a larger protein scaffold that in turn is designed to achieve optimal packing and interactions with the target protein. This approach solves two problems: (1) by beginning with a motif that is known to bind the target, the design immediately starts with some favorable interactions, and (2) the scaffold orientation against the target surface is guided by the motif itself. By using this information, the design is biased toward sampling only a small number of permissible docked configurations. Seeded protein–protein interface design strategies are indeed extremely powerful for creating novel protein binders, but the methods are also daunting for newcomers.

In this chapter, we describe a step-by-step workflow for the design of new protein binders based on motif grafting and "seeded" interface design. The majority of the protocols described can easily be run on a single personal computer, though large clusters and supercomputers will increase sampling and help find better solutions.

## 2 Materials (Required Software)

*ROSETTA*. The ROSETTA software suite includes algorithms for protein modeling and design [4]. ROSETTA is free for academic users and can be downloaded from: https://www.rosettacommons.org/software.

In the examples given here, ROSETTA was compiled and executed on a MacBook Pro with a 2.5 GHz quad-core Intel i7 processor. Basic knowledge of UNIX-style terminal commands is necessary.

For any design or structure prediction problem within ROSETTA, the potential energy is calculated using ROSETTA's energy function, which includes terms for attributes such as rotamer energies, van der Waals interactions, and hydrogen bonding, among others [5]; the process of applying the energy function to a given protein conformation is simply referred to as "scoring." As with free energy, a conformation or sequence with a lower energy in ROSETTA is more favorable. During protein structure prediction, the conformation of lowest energy is determined for a given amino acid sequence. During protein–protein interface design, the problem is reversed. Since the basic docked configuration of the binding partners is now known, the aim is to design the lowest energy sequence to stabilize the bound state of the two proteins.

*ROSETTA and RosettaScripts*. ROSETTA protocols are written in an XML-script format. The script is interpreted using the RosettaScripts parser, which is packaged within the ROSETTA suite [6]. Using a simple analogy, RosettaScripts protocols are like cooking recipes; they first define the ingredients (energy functions, task operations, filters, and movers) and then outline the protocol by which these are combined. RosettaScripts is easy to use, even for novices with minimal programming experience. Wiki-style documentation can be accessed at: https://www.rosettacommons.org/docs/latest/scripting_documentation/RosettaScripts/RosettaScripts.

This website provides an index of available operations and is an excellent resource when creating or modifying scripts.

*Important*: For the examples presented here, command lines contain the environment variable ${Rosetta}, which means the directory path in which ROSETTA is installed on the user's computer.

*Molecular Visualization*. A molecular graphics-viewing program is required. PyMol (Schrödinger, LLC) is recommended, as it has excellent and easy-to-use features for visualization, simple structural alignments, and even allows modifying proteins. A limited educational version (precompiled for several platforms) is available for free from: https://www.pymol.org/.

A full-featured open-source branch from SourceForge (Slashdot Media, requires compilation) is available at: http://sourceforge.net/projects/pymol/.

## 3 Methods

The workflow (Fig. 1) for computational interface design using motif grafting is comprised of the following steps:

1. Definition of the binding motif for seeded interface design.
2. Preparing a scaffold database.
3. Matching for putative scaffolds (i.e., motif grafting).
4. Sequence design.
5. Selection and improvement of designs.

### 3.1 Definition of the Binding Motif for Seeded Interface Design

To guide readers through each of these steps, we present the example of designing a protein binder for the estrogen receptor (ERα) based on a known peptide interaction. The crystal structure of ERα has been solved with a bound helical peptide from a transcriptional coactivator (PDB ID 1GWQ; Fig. 2) [7]. This natural protein–peptide complex provides an initial structural motif for seeded interface design. The bound peptide provides the core of the interface, and the design process involves transplanting/grafting the motif into alternative protein scaffolds, followed by design of neighboring residues close to the target protein surface, creating an extended interface for improved affinity and specificity.

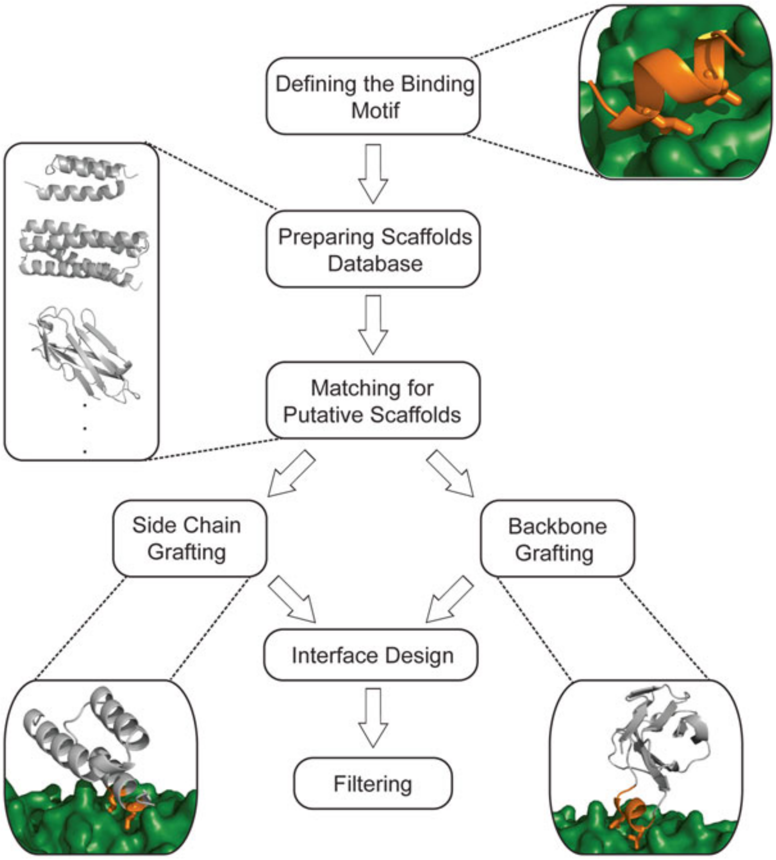
Defining the Binding Motif
Preparing Scaffolds Database
Matching for Putative Scaffolds
Side Chain Grafting
Backbone Grafting
Interface Design
Filtering

**Fig. 1** Workflow for seeded interface design. In the *inset panels*, the target protein surface is colored in *green*, the motif to be grafted in *orange*, and scaffolds are shown in *grey*

ERα is a steroid hormone-activated transcription factor that recruits coactivators to a target gene [8]. The ERα-coactivator interaction is established through a helical motif that bears the signature sequence LXXLL (where L is leucine and X is any amino acid), with the leucine residues (hot spots) binding a hydrophobic cleft on the ERα surface (Fig. 2b) [7]. In the following sections, we show how to graft the helical motif into a new protein scaffold. The assumptions guiding this design strategy are: (1) stabilization of the bound conformation of the LXXLL motif by embedding it within a stable scaffold reduces the entropic penalty of binding a flexible peptide, and (2) expanding the interfacial contact area can create new favorable interactions with the target. If successful, a design that combines these two factors can achieve an interaction with enhanced affinity and specificity.

First, the PDB of the protein–peptide complex is formatted for compatibility with ROSETTA and the structure is minimized (*see*

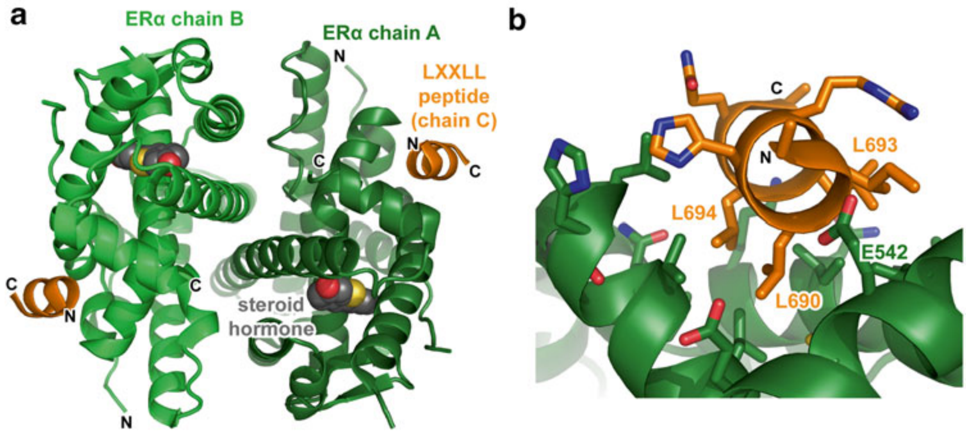

**Fig. 2** The ERα-LXXLL peptide complex. (**a**) The crystal structure of the ligand-binding domain of ERα (a dimer; two chains are shown in *light* and *dark green*) bound to the aroylbenzothiophene core of raloxifene (*grey spheres*) and a peptide (*orange*) spanning the helical LXXLL motif from the transcriptional coactivator TIF2 (PDB 1GWQ). PDB files of the motif (chain C) and target (chain A) are prepared. (**b**) The three conserved leucines of the LXXLL motif interact with a hydrophobic cavity on the ERα surface, while glu-542 of ERα caps the peptide's N-terminus

**Note 1** at the end for a detailed description on preparing input PDB files). Next, the structure is divided into two new PDB files, referred to as the "context" and "motif." The "context" file contains the target structure (i.e., ERα; only chain A of PDB ID 1GWQ), while the "motif" file contains the LXXLL peptide (chain C of PDB ID 1GWQ). In different scenarios, the motif could also be a small segment of a much larger protein, for example, an interacting loop extracted from an antibody–antigen structure.

### 3.2 Preparing a Scaffold Database

To prepare an inclusive scaffold database that can be searched for a variety of structural motifs, we downloaded 1519 structures from the PDB (www.rcsb.org) based on the following four criteria: (1) crystal structures with high-resolution x-ray diffraction data (<2.5 Å), (2) the proteins had been reported to be expressable in *E. coli* (this simplifies later experimental characterization), (3) a single protein chain in the asymmetric unit (MotifGraft only works with monomeric scaffolds as grafting targets), and (4) no bound ligands or modified residues. The scaffold PDB files were formatted for ROSETTA and subjected to an energy minimization step (*see* **Note 2**).

In some circumstances, a focused scaffold library may produce more useful matches. For our particular example, the peptide that seeds interface design has an α-helical conformation. Therefore, we also prepared a small focused scaffold library of 28 helical proteins.

### 3.3 Matching for Putative Scaffolds

The scaffold library is computationally scanned for possible graft sites. If the motif and scaffold backbones superimpose with very low root mean squared deviation (RMSD < 0.5 A), then only hot spot side chains need be transplanted from the motif to the corresponding positions in the matching site of the scaffold [9, 10]. This is known as “side chain grafting.” Subsequently, surrounding residues on the scaffold surface that are in contact with the target are designed for favorable interactions [3]. We suggest that side chain grafting should be attempted first, as it makes the minimal number of changes to the scaffold, increasing the chances of obtaining correctly folded designs during experimental validation. However, often side chain grafting is not possible because the motif and scaffold structures are too dissimilar. In these cases, even though the motif and scaffold may have very different structures, it is still possible to use an alternative method known as “backbone grafting” [11, 12].

During backbone grafting, the algorithm looks for segments of the scaffold backbone that align closely to the termini of the motif (both N- and C-terminal sides), and then the scaffold segment between these alignment points is replaced by the motif. This technique is extremely versatile, for example, a loop in the scaffold might be replaced by a peptide motif with different secondary structure, or even with a different amino acid length. Since the changes to the scaffold structure following backbone grafting can disrupt the overall fold, it is important to design the hydrophobic core to support the new backbone structure of the scaffold, followed by design of the protein–protein interface. The backbone grafting procedure often introduces many mutations to the scaffold, requiring careful filtering of designs to select those that present quality interfaces and high stability of the new scaffold.

The flow chart in Fig. 1 details the steps involved for both design strategies. We begin by describing side chain grafting, followed by backbone grafting.

### 3.4 Sequence Design

#### 3.4.1 Side Chain Grafting with RosettaScripts

Motif matching and interface design are distinct conceptual steps, but due to the flexibility of the RosettaScripts framework, both can be included in a single computational step. First, a list is generated containing all PDB files within the scaffold database:

```
#> ls -1 scaffolds_directory/*.pdb > scaffolds.list
```

Then RosettaScripts is executed using the following command:

```
#> ${Rosetta}/main/src/bin/rosetta_scripts -database ${Rosetta}/main/database/ -l scaffolds.list -use_input_sc -ex1 -ex2 -nstruct 1 -parser:protocol MotifGraft_sc.xml
```

The command line includes several important options. First, the location of the ROSETTA database must be specified using `-database`. Option `-l scaffolds.list` specifies the input list of scaffold PDB files. (Option `-s scaffold.pdb` would specify a single PDB file.) The options `-ex1` and `-ex2` allow ROSETTA to explore additional side chain rotamers, and `-use_input_sc` means that rotamers in the input structure are included in the rotamer library. Finally, option `-nstruct 1` means that the design script will be launched once per input scaffold. This can be increased if the user wishes to filter through more designs, but requires usage of the MultiplePoseMover in the XML script (for further information see RosettaScripts documentation).

In the case of grafting by side chain replacement, it took less than an hour to scan through the focused scaffold library of 28 helical proteins on a laptop computer and generate 23 designs. (Since several steps in the design process are stochastic, the number of results that pass the filters might vary if the protocol is re-executed.). The XML file MotifGraft_sc.xml reads as follows:

```
<ROSETTASCRIPTS>
<TASKOPERATIONS>
  <ProteinInterfaceDesign   name=pido   repack_chain1=1
  repack_chain2=1    design_chain1=0    design_chain2=1
  interface_distance_cutoff=8.0/>
  <OperateOnCertainResidues name="hotspot_repack">
    <RestrictToRepackingRLT/>
    <ResiduePDBInfoHasLabel property="HOTSPOT"/>
  </OperateOnCertainResidues>
</TASKOPERATIONS>
<SCOREFXNS>
</SCOREFXNS>
<FILTERS>
  <Ddg name=ddg confidence=0/>
  <BuriedUnsatHbonds name=unsat confidence=0/>
  <ShapeComplementarity name=Sc confidence=0/>
</FILTERS>
<MOVERS>
  <MotifGraft  name="motif_grafting"  context_structure=
  "context.pdb" motif_structure="motif.pdb" RMSD_toler-
  ance="0.3" NC_points_RMSD_tolerance="0.5" clash_score_
  cutoff="5"   clash_test_residue="GLY"   hotspots="3:7"
  combinatory_fragment_size_delta="2:2"  full_motif_bb_
  alignment="1"graft_only_hotspots_by_replacement="1"
  revert_graft_to_native_sequence="1"/>
  <build_Ala_pose  name=ala_pose  partner1=0  partner2=1
  interface_cutoff_distance=8.0
  task_operations=hotspot_repack/>
  <Prepack name=ppk jump_number=0/>
  <PackRotamersMover    name=design    task_operations=
  hotspot_repack,pido/>
  <MinMover name=rb_min bb=0 chi=1 jump=1/>
</MOVERS>
```

```
<PROTOCOLS>
  <Add mover_name=motif_grafting/>
  <Add mover_name=ala_pose/>
  <Add mover_name=ppk/>
  <Add mover_name=design/>
  <Add mover_name=rb_min/>
  <Add mover_name=design/>
  <Add filter_name=unsat/>
  <Add filter_name=ddg/>
  <Add filter_name=Sc/>
</PROTOCOLS>
</ROSETTASCRIPTS>
```

Within the XML file, the user may first specify which score/energy function to use from the ROSETTA database or reweight specific score terms; if no score function is defined, the default is used (currently "talaris2013," but this will likely change in future ROSETTA releases). Next, task operations define which residues can be altered. The ProteinInterfaceDesign task operation restricts design to residues of chain 2 (the scaffold) within 8 Å of the interface, while target residues within 8 Å of the interface may repack to alternative low-energy rotamers. By default, the design of nonnative prolines, glycines, and cysteines (which can have important structural consequences) is forbidden. The second task operation, RestrictToRepackingRLT, prevents the two grafted hot spot leucines from being mutated in later design steps, though they can repack to alternative rotamers. (For polar hot spot residues, alternative rotamers would disrupt hydrogen-bonding networks, and we would advise using the more restrictive task operation PreventRepackingRLT, which prevents both design and repacking.) The MotifGraft mover (described below) keeps track of which residues correspond to the target, scaffold, or motif and which critical side chains are grafted. These are labeled CONTEXT, SCAFFOLD, MOTIF, and HOTSPOT, respectively. These residue classes are then available for task operations, as used here. The details for these task operations are given on the wiki website: https://www.rosettacommons.org/docs/latest/scripting_documentation/RosettaScripts/TaskOperations/taskoperations_pages/OperateOnCertainResiduesOperation.

Movers dictate how the protein complex is manipulated, such as sequence design, side chain and backbone minimization, or rigid-body docking. The protocol begins with the MotifGraft mover, which searches for alignments between the scaffold and motif that do not produce steric clashes with the target structure. The MotifGraft mover has many options. First, the names of the PDB files for the target (context_structure) and motif (motif_structure) must be specified. The option RMSD_tolerance sets the maximum RMSD allowed between the motif and scaffold alignment. For side chain grafting, the motif should closely match the scaffold segment it is aligned with, so that the backbones are virtually superimposable. In this XML script, the RMSD tolerance was set to 0.3 Å (maximum recommended is ~0.5 Å). The option NC_points_RMSD_tolerance sets the maximum RMSD allowed between the N-/C-termini of the motif and scaffold graft site (recommended 0.5 Å). Once the scaffold has been aligned, the configuration of the system must be checked for clashes. After it is grafted, the motif cannot clash with other parts of the scaffold (this is not an issue for side chain grafting when the motif closely matches a native structural region within the scaffold, but is of serious concern when performing backbone grafting).

In addition, the orientation of the scaffold when aligned with the motif cannot clash with the target surface. Since residues can be designed to smaller amino acids in later steps, clashes are checked after first mutating the motif to small amino acids, such as alanine or glycine (using option `clash_test_residue="GLY"` in this XML script). All the atomic clashes are computed, and if the score is above the clash_score_cutoff, the graft fails and an alternative alignment in the scaffold is attempted (it is recommended to set the clash_score_cutoff at ≤ 5). The options `full_motif_bb_alignment="1"` and `graft_only_hotspots_by_replacement="1"` indicate that side chain grafting is being performed. Option `hotspots="3:7"` defines which positions in the motif PDB correspond to the two leucine hot spots of the LXXLL peptide. Additional hot spots are each separated by colons. Option `combinatory_fragment_size_delta="2:2"` indicates by how many amino acids the motif may be shortened at each terminus (N-terminus:C-terminus), i.e., whether the full motif must align ("0:0") or only a partial fragment. Here, the algorithm will attempt to match the full-length motif, as well as each motif fragment shorter by up to two residues at one or both termini. The final option, `revert_graft_to_native_sequence="1"`, means that after the motif has been placed into the scaffold, all residues except for the hot spots are reverted back to their native identities. Therefore, only the two hot spot amino acids are effectively transferred as changes to the scaffold sequence.

After side chain grafting, the protocol continues by replacing scaffold side chains within 8 Å of the target with alanine using the build_Ala_pose mover. Task operations prevent the hot spots from changing. Side chains are now repacked with the Prepack mover. During this step, target protein residues that sterically clash with the scaffold have the opportunity to find alternative, non-clashing rotamers. Next, the interface surrounding the grafted hot spots is designed using the PackRotamersMover. Task operations ensure that hot spot and target residues can only change rotamer conformations, whereas scaffold residues within 8 Å of the target surface are available for design. Side chains and rigid-body orientations of the designed complex are then minimized with MinMover, followed by a second round of design.

Multiple rounds of minimization and design are recommended as they may improve results. Further details about movers can be found at: https://www.rosettacommons.org/docs/latest/scripting_documentation/RosettaScripts/Movers/Movers-RosettaScripts.

Finally, three filters are used to assess the designs' structural features: binding energy (ΔΔG), interface shape complementarity, and buried unsatisfied hydrogen-bonding atoms at the interface. In this example XML script, each filter is assigned a confidence of 0, such that all designs will pass. Rather than acting to terminate design calculations, these filters are instead being used to report interface quality. Based on these reported values, the user can determine which are the best designs of the pool. A full list of available filters can be found at: https://www.rosettacommons.org/docs/latest/scripting_documentation/RosettaScripts/Filters/Filters-RosettaScripts.

Some examples of the designs generated by the aforementioned script are shown in Fig. 3. XML scripting is amenable to rapid protocol modifications, and users are encouraged to attempt their own variations of the protocols. The RosettaScripts online documentation is an excellent resource to understand the functionality that different options provide.

### *3.4.2 Backbone Grafting with RosettaScripts*

Using the same motif and target PDB files described above, we present an example XML script that scans scaffolds for potential backbone graft sites and subsequent design. The script can be executed as follows:

```
#> ${Rosetta}/main/source/bin/rosetta_scripts.macosclang-
release -database ${Rosetta}/main/database/ -l scaf-
folds.list -use_input_sc -nstruct 1 -parser:protocol
MotifGraft_bb.xml
```

The XML script reads:

```
<ROSETTASCRIPTS>
<TASKOPERATIONS>
  <ProteinInterfaceDesign name=pido_far interface_distance
  _cutoff=15.0/>
  <ProteinInterfaceDesign name=pido_med interface_distance_
  cutoff=12.0/>
  <ProteinInterfaceDesign name=pido_near interface_distance_
  cutoff=8.0/>
  <OperateOnCertainResidues name="hotspot_repack">
    <RestrictToRepackingRLT/>
    <ResiduePDBInfoHasLabel property="HOTSPOT"/>
  </OperateOnCertainResidues>
  <SelectBySASA name=core mode="sc" state="bound" probe_
  radius=2.2 core_asa=0 surface_asa=30 core=1 bound-
  ary=0 surface=0/>
  <SelectBySASA name=core_and_boundary mode="sc" state=
  "bound" probe_radius=2.2 core_asa=0 surface_asa=30
  core=1 boundary=1 surface=0/>
</TASKOPERATIONS>
```

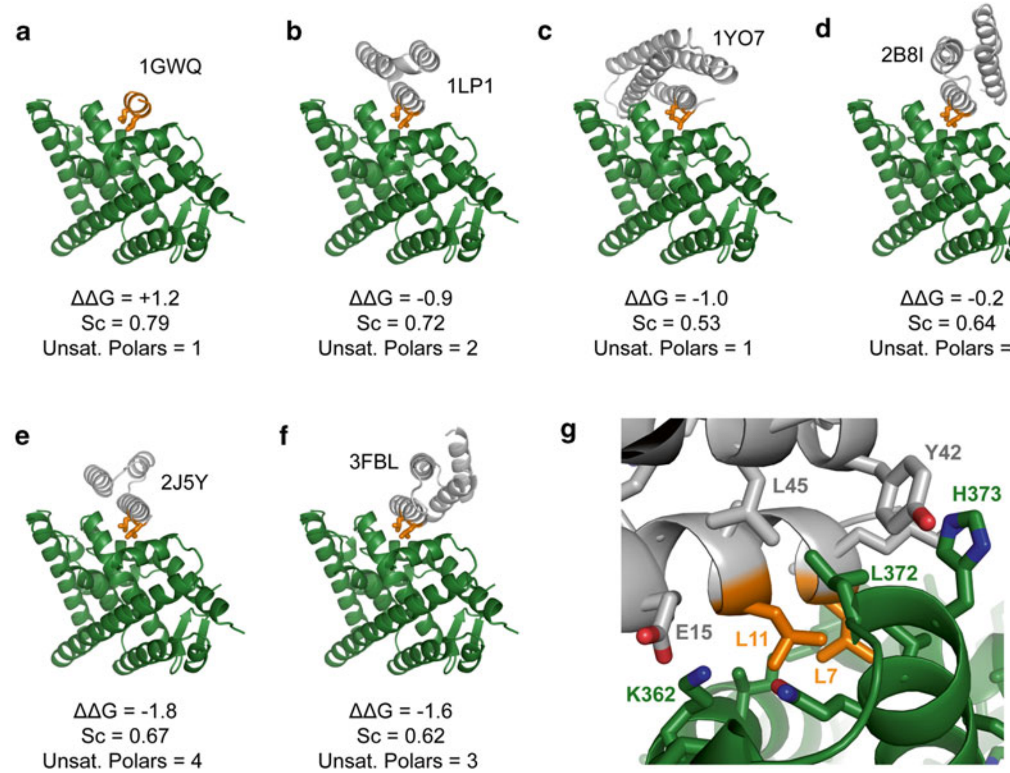

**Fig. 3** Examples of designs generated by side chain grafting. (**a**) The crystal structure (PDB 1GWQ) of a LXXLL coactivator motif (*orange*) bound to ERα (*green*). Only chains A (ERα; the target) and C (LXXLL motif) are considered. The structure was energy minimized with ROSETTA and the interface was scored. (**b**–**f**) Five different designs generated by side chain grafting using the XML script described here. The scaffolds (*grey*; PDB codes indicated in the figure) are all helical bundle proteins. The grafted leucine hot spot residues (L690 and L694 in Fig. 2) are colored in *orange*. (**g**) The interface of the design in panel (**b**) is shown in greater detail. Designed interactions around the hot spots include hydrophobic contacts from L45, aromatic stacking between designed residue Y42 and target residue H373, and a saltbridge from E15 to K362

```
<FILTERS>
  <Ddg name=ddg confidence=0/>
  <BuriedUnsatHbonds name=unsat confidence=0/>
  <ShapeComplementarity name=Sc confidence=0/>
</FILTERS>
<MOVERS>
  <MotifGraft name="motif_grafting" context_structure=
  "context.pdb" motif_structure="motif.pdb" RMSD_toler-
  ance="1.0"   NC_points_RMSD_tolerance="1.0"   clash_
  score_cutoff="5"  clash_test_residue="GLY"  hotspots=
  "3:7"combinatory_fragment_size_delta="2:2"  max_frag-
  ment_replacement_size_delta="-8:8" full_motif_bb_align-
  ment="0" graft_only_hotspots_by_replacement="0"/>
  <build_Ala_pose  name=ala_pose  partner1=0  partner2=1
  interface_cutoff_distance=8.0  task_operations=hotspot_
  repack/>
  <Prepack name=ppk jump_number=0/>
  <PackRotamersMover name=design_core task_operations=
  hotspot_repack,pido_far,core/>
  <PackRotamersMover name=design_boundary task_operations=
  hotspot_repack,pido_med,core_and_boundary/>
  <PackRotamersMover name=design_interface task_operations=
  hotspot_repack,pido_near/>
  <MinMover name=sc_min bb=0 chi=1 jump=0/>
</MOVERS>
<PROTOCOLS>
  <Add mover_name=motif_grafting/>
  <Add mover_name=ala_pose/>
  <Add mover_name=ppk/>
  <Add mover_name=design_core/>
  <Add mover_name=design_boundary/>
  <Add mover_name=design_interface/>
  <Add mover_name=sc_min/>
  <Add filter_name=unsat/>
  <Add filter_name=ddg/>
  <Add filter_name=Sc/>
</PROTOCOLS>
</ROSETTASCRIPTS>
```

The first mover called in the protocols section of the XML script is MotifGraft. As with side chain grafting, options `context_structure` and `motif_structure` specify the target and motif PDB files, respectively. The `RMSD_tolerance` and `NC_points_RMSD_tolerance` are both set at 1.0 Å (the maximum recommended is 1.5 Å); during backbone grafting, these options set the maximum allowed RMSD between the motif termini and the backbone graft sites in the scaffold. A lower RMSD tolerance will enforce a better match between the motif termini and scaffold backbone, giving better results, though at the expense of more solutions. The options for `clash_test_residue, clash_score_cutoff, hotspots` and `combinatory_fragment_size_delta` are set the same as for side chain grafting. However, for backbone grafting options `full_motif_bb_alignment` and `graft_only_hotspots_by_replacement` are both turned off (i.e., set to "0"). A new option is now used; `max_fragment_replacement_size_delta="-8:8"` sets the minimum and maximum sizes of the scaffold segment that can be replaced by the motif (i.e., the resulting scaffold can vary from eight residues shorter up to eight residues longer than the original scaffold).

The protocol continues by calling a mover to mutate scaffold residues at the interface to alanine. Next, rotamers are minimized with the Prepack mover, followed by three design steps using PackRotamersMover. The first design step is restricted to scaffold residues within the hydrophobic core up to 15 Å away from the interface. Since the grafted motif is potentially very different from the scaffold segment it replaced, design of the core is necessary to stabilize the new structure. Two task operations define which residues can be designed: (1) the ProteinInterfaceDesign task operation permits design to chain 2 (the scaffold) within a distance threshold of the interface, and (2) the SelectBySASA task operation defines core, boundary, and surface residues based on solvent-accessible surface area and turns their design on or off. The second design step is restricted to 12 Å from the interface but now allows the design of core and "boundary" (i.e., partially buried) amino acids. Again, task operations define the residues for design. The third design step is now focused on optimizing all scaffold residues 8 Å from the target surface. A task operation prevents the grafted hot spot leucine residues from mutating at any stage. The final mover is a side chain minimization.

The protocol finishes with three filters to report on interface quality: the calculated binding energy, number of buried unsatisfied hydrogen-bonding atoms, and shape complementarity. Within 3 h on a laptop computer, over 200 scaffolds in the library were scanned for potential graft sites, and nearly as many designs were generated. In many of the designed proteins, helical segments of the scaffolds were swapped with the helical motif. However, in other designs, a non-helical scaffold segment was replaced; some examples are shown in Fig. 4.

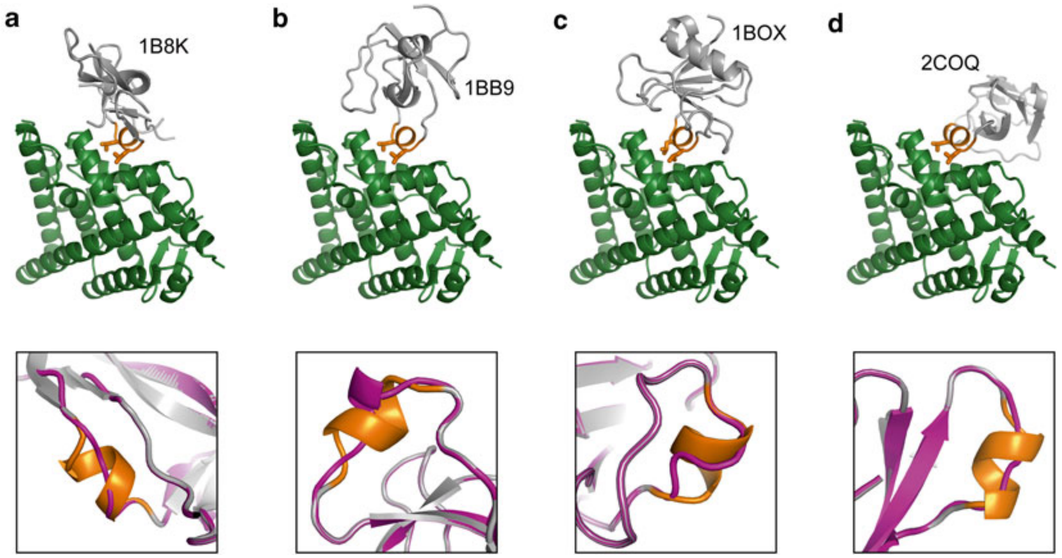

**Fig. 4** Examples of designs generated by backbone grafting. (**a**–**d**) In the *upper* images, the target ERα is shown in *green*, the scaffold in *grey*, and the grafted motif in *orange*. The scaffold PDB is labeled. In the *lower* images, the designed proteins (scaffold and motif regions are in *grey* and *orange*, respectively) are superimposed with the original scaffold PDBs in *magenta*. Notice that scaffold loops of very different lengths and conformations were replaced with the helical motif

### 3.5 Selection of Designs and Optimization

To date, no computational method has been developed that can predict with perfect accuracy which designs will be functional when challenged experimentally [13]. Therefore, it is wise to proceed with designed sequences that present good metrics by multiple criteria. Designs are initially filtered based on calculated metrics for interface quality, including a favorable binding energy ($\Delta\Delta G < 0$ ROSETTA energy units, ideally the energy should be lower than the native interface from which the motif was taken), high shape complementarity ($Sc > 0.65$), and a low number of buried unsatisfied hydrogen-bonding atoms. In the XML scripts above, these filters report to a score file and will also be appended at the end of any ROSETTA output PDBs.

Once a set of designs have been selected based on the calculated metrics, it is important to perform human-guided inspection of the designed structures. There are many qualities of interfaces that are apparent to structural biologists that are not captured in standard metrics. Two common defects in ROSETTA-designed structures that are very important to avoid are buried charged residues and under-packed interfaces dominated by alanine residues (Fig. 5).

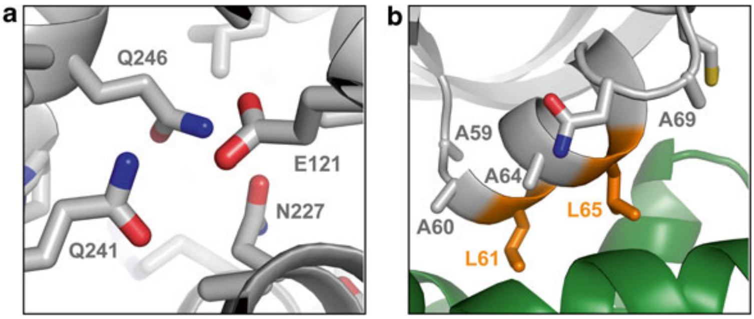

**Fig. 5** Common defects in ROSETTA-designed protein binders. (**a**) After backbone grafting, the hydrophobic core of scaffold 1A0P (*grey*) was designed to support the motif. Polar and charged residues (*labeled*) were designed within the core; however, native proteins nearly always have hydrophobic cores. (**b**) Scaffold (PDB 2B29) is shown in *grey*, while the grafted leucines are in *orange* and the target ERα is *green*. The majority of designed scaffold residues at the interface (*grey sticks*) are alanines. Interfaces dominated by alanine can achieve low energies; alanine is a small hydrophobic residue that will not clash with the target surface and is therefore the "default" residue when specific interactions cannot be designed. These interfaces lack hydrogen-bonding networks and are generally under-packed

### 3.5.1 Reverting Designed Mutations Back to Native

It is also important to consider whether the designed scaffold will fold to its intended structure; having a spectacular interface on a computational model is irrelevant if the protein cannot fold in an experimental setting. This is particularly problematic for designed interfaces that have a large surface area dominated by hydrophobic residues. It is generally assumed that the probability of a designed sequence properly folding is inversely correlated with the number of mutations imposed on the scaffold during the design process. Therefore, it is beneficial to be conservative and make as few mutations as possible by reverting residues back to their native identities in a post-design stage. The ROSETTA application "revert_design_to_native" [2] can be used for this task; it goes through each mutated position in the scaffold, reverts to the native amino acid, and computes the change in binding energy. If the native residue scores similarly to the designed residue, then it may be safer to revert back to the native amino acid. The revert_design_to_native application requires two input PDBs: the designed PDB (containing the target (chain A) bound to the designed scaffold (chain B)) and a reference PDB that contains the target together with the native scaffold. To determine which residues have been mutated, the application sequentially compares each amino acid between the design and reference PDBs; this means the application can only be applied to designs from side chain grafting in which the two PDB files have the same number of residues. The reference PDB is easily generated by concatenating the target (context.pdb) with the scaffold PDB using the cat command:

```
#> cat context.pdb scaffold.pdb >nativecplx.pdb
```

Revert_design_to_native is run with the following command:

```
#>  ${Rosetta}/main/source/bin/revert_design_to_native.
macosclangrelease    -revert_app:wt    nativecplx.pdb
-revert_app:design design.pdb -ex1 -ex2 -use_input_sc
-database ${Rosetta}/main/database/
```

### 3.5.2 Manually Adjusting Designs Using FoldIt

If necessary, the designed structures may be subjected to human-guided optimization. The user may wish to correct a number of frequent problematic features in ROSETTA designs, such as hydrophobic residues at the water-exposed interface edge, revert designed residues back to their native identities, mutate buried charged residues to hydrophobics, etc. There are no hard rules for manually improving designs; it is simply a matter of the designer's preference and experience. FoldIt is an excellent computational tool to perform this human-guided optimization [14]. It combines a graphic front end with molecular visualization together with many basic tools such as sequence design, rotamer repacking, and minimization (though often with creative names like "Shake" and "Wiggle"). FoldIt was developed as a protein folding and design game, bringing the advantages of crowdsourcing to solve structural biology problems [14]. The stand-alone version of FoldIt gives immediate visual and ROSETTA energy feedback, helping the user decide if any further mutations to the designed protein are warranted. The license for FoldIt Standalone is available from http://c4c.uwc4c.com/express_license_technologies/foldit, and directions will then be provided for downloading the software.

#### 3.5.3 *Filtering Designs Based on Folding Probability*

Designs from backbone grafting require extra attention, as the engineering of a protein core to support the grafted motif can be challenging. Many designed sequences will not fold correctly when experimentally tested. We have found structure prediction to be a powerful filter; the designed amino acid sequences when subjected to structure prediction calculations should yield similar structures to the designed models [3]. If structure prediction returns an alternative conformation, or fails to converge on an energy minimum in a conformational landscape, then it is unlikely that the designed sequence will correctly fold. However, structure prediction is computationally expensive and not accessible on a large scale to most biochemists. Further, this evaluation method is only useful if the original scaffold sequence correctly returns the native structure; for many natural proteins, structure prediction methods are not yet able to accurately predict the known structure. Instead, designs can be relaxed with ROSETTA to determine if the designed conformation is "stable." If the designed structural model drifts, it is unlikely to occupy a low-energy conformation at the bottom of an energy funnel, and the design should either be rejected or improved using information derived from the relaxed ensemble, from which one can identify cavities and alternative conformations that should be eliminated by additional design steps. To apply this filter, first extract chain B (the designed protein) from the PDB files of the designed complexes:

```
#> for i in *.pdb; do grep " B " $i >$i.chainB; done
#> ls -1 *.chainB >monomers.list
```

Next, the designed monomers are relaxed and the RMSD to the starting structure is determined:

```
#> ${Rosetta}/main/source/bin/rosetta_scripts.macosclangrelease -database ${Rosetta}/main/database/ -l monomers.list -use_input_sc -nstruct 1 -parser:protocol fastrelax.xml
<ROSETTASCRIPTS>
<MOVERS>
  <FastRelax name=fstrlx repeats=4/>
</MOVERS>
<FILTERS>
  <Geometry name=omega omega=150 cart_bonded=100
  confidence=0/>
  <CavityVolume name=cav_vol confidence=0/>
  <Rmsd name=rmsd confidence=0 superimpose=1/>
</FILTERS>
<PROTOCOLS>
```

```
  <Add filter_name=omega/>
  <Add filter_name=cav_vol/>
  <Add mover_name=fstrlx/>
  <Add filter_name=rmsd/>
</PROTOCOLS>
</ROSETTASCRIPTS>
```

The RMSD will be low if the designed protein conformation is stable (typically ≤ 1 Å). This XML script also reports two other useful metrics prior to relaxation. The Geometry filter checks that backbone omega angles are above a defined cutoff (except for *cis*-prolines, omega angles should be close to 180°) and that Cartesian space bond angles and lengths are close to ideal (decrease the cart_bonded penalty score for a more stringent filter). The geometry at the junction points where the motif is grafted can be particularly poor, and in such cases the cart_bonded penalty score will be flagged as high and the omega angle as too low in the log report. The CavityVolume filter measures the total cavity volume in Å$^3$. This will be higher for bigger proteins and therefore should not be used as a hard filter, but any outliers with exceptionally high values likely have underpacked cores.

### 3.6 Experimental Validation

Despite notable advances, computational protein design has only modest success rates at the stage of experimental characterization. Hence, it is essential to have a robust and rapid experimental assay for evaluating designs. Library display methods are ideally suited to screening many designs individually or simultaneously within a mixed pool [3], and as the cost of DNA synthesis has plummeted, it is possible to screen hundreds to thousands of designs within a reasonable budget. Often initial computational designs present low affinities to the desired targets and must be optimized by targeted mutagenesis or directed evolution [1–3, 12, 15]. Experimental methods should be carefully considered before embarking on any protein design project.

### 3.7 Concluding Remarks

Computational design of protein–protein interactions is poised to make spectacular advancements. Fast computers, affordable DNA synthesis, and the development of tools like ROSETTA have coalesced in the past few years, such that computational design methodologies are now accessible to a wider community without requiring supercomputers or advanced programming skills. Here, we have outlined general methods for seeded interface design and encouraged readers to create new protocols tailored to their problems. Proteins made to order, once deemed science fiction, are rapidly becoming a reality.

## 4 Notes

1. *Formatting PDB files.* PDB files must be correctly formatted for compatibility with ROSETTA. All heteroatoms, including water molecules, should be removed. In ROSETTA "TER" statements designate different proteins in a complex, and therefore any "TER" statements within a single protein chain must be removed, such as those that are used to mark regions of missing density. While these modifications can be made in a text editor, a large number of PDB files can easily be prepared with the following UNIX command:

   ```
   #> for i in *.pdb; do grep "ATOM " $i >$i.atoms; done
   ```

   This will go through all PDB files within the directory, search for all lines containing the string "ATOM", and print these lines to a new file with suffix atoms.

2. *ROSETTA energy minimization of crystallographic structures.* It may be advantageous to perform energy minimization of the structures within the ROSETTA energy function prior to matching and design. Structures from experimental data often have residues with high (i.e., energetically unfavorable) energy due to minor clashes or "imperfections," and these may be inappropriately designed by ROSETTA to alternative amino acids. This is especially problematic for backbone grafting and may lead to unnecessary sequence design of residues that should remain unchanged. Energy minimization of input PDBs generally resolves this issue. However, it is important that structures do not drift too far during the minimization protocol; after all, the original PDB files are determined from real experimental data, whereas a minimized structure will only be as real as the energy function is accurate. To perform this step, we suggest two computational protocols. First, structures can be minimized using the constrained fast relaxation protocol. To minimize a single PDB file, use option -s file.pdb in the command line. To relax all PDB files within a directory, create a list first:

   ```
   #> ls -1 *.pdb >pdb_files.list
   #> ${Rosetta}/main/source/bin/relax.macosclangrelease
   -database ${Rosetta}/main/database/ -ignore_unrecog-
   nized_res -relax:constrain_relax_to_start_coords -ex1
   -ex2 -use_input_sc -l pdb_files.list
   ```

   Alternatively, structures can be minimized using RosettaScripts. A command line and example XML script are:

   ```
   #> ${Rosetta}/main/source/bin/rosetta_scripts.maco-
   sclangrelease -database ${Rosetta}/main/database/ -l
   pdb_files.list -use_input_sc -ex1 -ex2 -parser:protocol
   ppk_min.xml
   ```

Contents of ppk_min.xml:

```
<ROSETTASCRIPTS>
<FILTERS>
  <Rmsd name=rmsd threshold=1.5 superimpose=1/>
</FILTERS>
<MOVERS>
  <Prepack name=ppk jump_number=0/>
  <MinMover name=sc_bb_min bb=1 chi=1/>
</MOVERS>
<PROTOCOLS>
  <Add mover_name=ppk/>
  <Add mover_name=sc_bb_min/>
  <Add mover_name=ppk/>
  <Add mover_name=sc_bb_min/>
  <Add filter_name=rmsd/>
</PROTOCOLS>
```

In this XML script, there are two rounds of rotamer repacking and side chain/backbone minimization using the movers Prepack and MinMover. The "Rmsd" filter superimposes the minimized structure with the input PDB file; if the two differ by over 1.5 Å, then the structure is rejected and ROSETTA proceeds to the next scaffold in the list. The reasons why a structure is "unstable" during energy minimization and rejected may include inaccuracies in the ROSETTA energy function or regions of poor quality in the crystallographic models. For instance, in our initial scaffold library, we found that from 1519 protein structures, only 1419 fulfilled the filtering criteria and were included in the library to perform the modeling examples described in this manuscript.

## References

1. Procko E, Hedman R, Hamilton K et al (2013) Computational design of a protein-based enzyme inhibitor. J Mol Biol 425:3563–3575. doi:10.1016/j.jmb.2013.06.035
2. Fleishman SJ, Whitehead TA, Ekiert DC et al (2011) Computational design of proteins targeting the conserved stem region of influenza hemagglutinin. Science 332:816–821. doi:10.1126/science.1202617
3. Procko E, Berguig GY, Shen BW et al (2014) A computationally designed inhibitor of an Epstein-Barr viral Bcl-2 protein induces apoptosis in infected cells. Cell 157:1644–1656. doi:10.1016/j.cell.2014.04.034
4. Leaver-Fay A, Tyka M, Lewis SM et al (2011) ROSETTA3: an object-oriented software suite for the simulation and design of macromolecules. Methods Enzymol 487:545–574. doi:10.1016/B978-0-12-381270-4.00019-6
5. Das R, Baker D (2008) Macromolecular modeling with rosetta. Annu Rev Biochem 77:363–382. doi:10.1146/annurev.biochem.77.062906.171838
6. Fleishman SJ, Leaver-Fay A, Corn JE et al (2011) RosettaScripts: a scripting language interface to the Rosetta macromolecular modeling suite. PLoS ONE 6, e20161. doi:10.1371/journal.pone.0020161
7. Wärnmark A, Treuter E, Gustafsson J-A et al (2002) Interaction of transcriptional intermediary factor 2 nuclear receptor box peptides with the coactivator binding site of estrogen receptor alpha. J Biol Chem 277:21862–21868. doi:10.1074/jbc.M200764200
8. Savkur RS, Burris TP (2004) The coactivator LXXLL nuclear receptor recognition motif. J Pept Res 63:207–212. doi:10.1111/j.1399-3011.2004.00126.x
9. Ofek G, Guenaga FJ, Schief WR et al (2010) Elicitation of structure-specific antibodies by epitope scaffolds. Proc Natl Acad Sci U S A 107:17880–17887. doi:10.1073/pnas.1004728107

10. Correia BE, Ban Y-EA, Holmes MA et al (2010) Computational design of epitope-scaffolds allows induction of antibodies specific for a poorly immunogenic HIV vaccine epitope. Structure 18:1116–1126. doi:10.1016/j.str.2010.06.010
11. Azoitei ML, Ban Y-EA, Julien J-P et al (2012) Computational design of high-affinity epitope scaffolds by backbone grafting of a linear epitope. J Mol Biol 415:175–192. doi:10.1016/j.jmb.2011.10.003
12. Azoitei ML, Correia BE, Ban Y-EA et al (2011) Computation-guided backbone grafting of a discontinuous motif onto a protein scaffold. Science 334:373–376. doi:10.1126/science.1209368
13. Fleishman SJ, Whitehead TA, Strauch E-M et al (2011) Community-wide assessment of protein-interface modeling suggests improvements to design methodology. J Mol Biol 414:289–302. doi:10.1016/j.jmb.2011.09.031
14. Cooper S, Khatib F, Treuille A et al (2010) Predicting protein structures with a multiplayer online game. Nature 466:756–760. doi:10.1038/nature09304
15. Whitehead TA, Chevalier A, Song Y et al (2012) Optimization of affinity, specificity and function of designed influenza inhibitors using deep sequencing. Nat Biotechnol 30:543–548. doi:10.1038/nbt.2214